UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

Analysis of sensory data using Graph Signal Processing

Author
Ivan Salfati

Supervisor
Jose M. Barceló

Master in Innovation and Research in Informatics: Computer Networks and Distributed Systems

Defense Date: 29 June 2020

Abstract

Air pollution monitoring is an important topic that has been researched in the past few years thanks to the massive deployment of IoT platforms, as it affects the lives of both children and adults, and it kills millions of people worldwide every year. A new framework of tools called Graph Signal Processing was presented recently and it allows, among other things, the ability to predict data on a node that belongs to a network of sensors using both the data itself and the topology of the graph, which is based on the Laplacian matrix.

This thesis is a comparative study on different prediction techniques for pollutant signals, such as Linear Combination, Multiple Linear Regression and GSP and it presents the results of all three methods in different scenarios, using RMSE and R2 indicators, and focusing the efforts on the understanding of how different parameters (such as the distance between nodes) affect the performances of these new tools.

The results of the study show that pollutants O_3 and NO_2 are low-pass signals, and as the number of edges between nodes increases, GSP obtains a close performances to MLR. For PM_{10}, we conclude that is not a low-pass signal, and the performance of the indicators drop massively compared with the previous ones. Linear combination is the worst of all three and MLR has a stable performance during all the scenarios.

Contents

Chapter 1

Introduction

1.1 Background

In a world where air pollution kills an estimated seven million people worldwide every year it is important to check the levels of pollutants in the air. To do so, World Health Organization (WHO) [1] works with countries to monitor the air pollution levels and improve the air quality.

From smog hanging over cities to smoke and dust inside our homes, air pollution poses a major thread to health and climate. The combined effects causes millions of premature deaths every year, largely as a result of increased mortality from stroke, heart disease, lung cancer or acute respiratory infections.

More than 80% of the population living in urban areas that monitor air pollution are exposed to air quality levels that far exceed WHO guideline limits, with low and middle income countries suffering from the highest exposures.

The major outdoor pollution sources include vehicles, power generation, building's heating systems, incineration of agriculture/waste and industry, which produces ground-level ozone (O_3) and nitrogen dioxide (NO_2), [2][3] which are dangerous pollutants. Although policies and investments supporting cleaner transport, energy-efficient housing, power generation, industry and better municipal waste management can effectively reduce key sources of ambient air pollution.

Household pollution is important too, as may lead to a wide range of adverse health outcomes in both children and adults. Exposure to smoke from cooking or burning wood and coal in inefficient stoves or construc-

tion works produces a variety of health-damaging pollutants, including particulate matter of different sizes (PM_{10}, $PM_{2.5}$) [4], methane, carbon monoxide, etc.

With this in mind, and thanks to the growing interest in deploying Internet of Things (IoT) platforms to analyze air pollution, this thesis aims to analyze sensor signals and evaluate prediction techniques for different pollutants, namely O_3, NO_2 and PM_{10}, around the metropolitan area of Barcelona, Spain, using stationary stations provided by the *Generalitat de Catalunya* as nodes on a irregular graph using Graph Signal Processing (GSP).

1.2 Motivation

In the last few years, there has been a great research on analysis and data prediction such as [5][6][7], where a wide variety of methods are featured.

But it was around 2012-2014 when an interesting proposal called GSP was made. Several works explore and analyze these techniques, for example, the surveys of Ljubiša Stanković *et al.* [8][9][10], where the aim is to develop tools for processing data defined on irregular graph domains, extending Discrete Signal Processing (DPS) [11][12] to signal samples indexed by nodes of a graph. Other interesting surveys worth mentioning are [13] and [14].

At very high level, DSP and therefore GSP, study: 1) signals and their representations, 2) systems that process signals, 3) signal transforms and 4) sampling of signals.

Learning how these new tools work, in terms of the quality of the predicted data is key to compare them. To do so, we have indicators such as the Root Mean Square Error (RMSE) and the R-squared (R2). These two indicators will allow me to see how different parameters affect the prediction of the data, even if the data has been inferred with Gaussian noise while using GSP.

The motivation of using GSP comes from the opportunities that presents when it predicts data versus other well known techniques such as Multiple Linear Regression (MLR) which will be discussed later on in this document. On the other hand, studying such a new tools is an exciting topic which has a few research applied to monitoring air pollution in sensor networks.

1.3 Goals

In this Master Thesis, we will explore the use of GSP techniques in the prediction of sensoring data applied to the monitoring of air pollution in urban environments. For this purpose, we will consider a network of nodes that capture air pollution data, e.g. ozone (O_3), dioxide of nitrogen (NO_2) or particulate matter (PM_{10}) and see the potential of GPS techniques to predict data on specifics points of the network. Here we will build a graph and explore:

- What is the performance of GSP when reconstructing signals over other techniques.
- How do the different GSP parameters (such as the distance between nodes) affect the signal reconstruction.
- How a malfunctioning station affects the performance of GSP techniques.

In the end, we want to explore whether GSP might be an interesting technique for predicting sensor values using a network of nodes.

Chapter 2

State of the Art

2.1 Sensor Calibration

Recently, signal processing techniques have been applied to studies related to calibration of low cost sensors. Mueller *et al.* [15] and Liu *et al.* [16] have shown that, in order to calibrate air pollution sensors, it is necessary to have an array of them.

The idea behind the array of sensors consists in measuring all the cross-sensitivities to compensate for all interfering pollutants and environmental conditions [17][18][19]. For example, in order to calibrate NO_2 sensors, NO_2, O_3, temperature and relative humidity are also measured.

Several methods and algorithms have been studied, either establishing a linear relationship between measured gas concentrations and the sensor responses, or, as a more sophisticated calibration functions, those who include multiple corrections of several gaseous and physical variables to limit the impact of external interference.

Simpler methods include deterministic correction of sensor response to solve the problem of gaseous interfering compounds, for example, subtracting the O_3 interference from a NO_2 electrochemical sensor, that is a well known technique for simultaneously measure O_3 and NO_2 [20][21].

Complex methods and algorithms use data generated by metal oxides sensors (MOx) operated with temperature cycles to improve ozone sensitivity. Other possible calibration approaches assume a distributed network of nodes.

To keep it simple, whenever the sensor responses has a linear behavior with respect to the reference data, MLR is used for calibrating the sen-

sors. Nonetheless, if it is not linear, methods like Gaussian processes are used [22].

2.2 Air Pollution Monitoring Projects

There are a lot of countries that monitor the air pollution of their cities, in order to find anomalies and/or to create a record of the measurements.

For example, the project CAPTOR [23] was a project funded by the European Union between 2016 and 2019, where partners coming from Spain, Austria, Italy and France, addressed, in general, the air pollution problem, more specifically, they actively engaged citizens, scientists and farmer's unions, in a collaborative monitoring of O_3 pollution in Europe.

It was also designed to leverage local networks of volunteers that could deploy measuring devices on their houses, raising awareness of the pollution problem, becoming more in the public domain. On a volunteer basis, 20 individuals over Catalonia lent their balconies, porches and windows so that a group of researches could install measurement DIY stations that collected information about the air quality.

Those DIY nodes had two versions. The first one, "Captor II" was based on MOx sensors using an Arduino Yun board. The second one, "Captor III", had a shield for electrochemical sensors using a Rasberry Pi board. More details of this project can be found here [24]. Additionally, the data collected can also be found in [25][26][27].

The Statistical Analysis of Networks and Systems research group (SANS) [28] of the Computer Architecture Department at the Polytechnic University of Catalonia coordinated the part of the project in Spain.

There are other projects like CAPTOR, as we can see in the work done by Sami Kaivonen and Edith C.H. Ngai [29] (which is part of the GreenIoT project [30]) where they present an experimental study on real-time air pollution wireless sensors on public transport vehicles on the city of Uppsala, Sweden, through the deployment of low-cost wireless sensors.

The study show that it is possible to obtain a more fine-grained real-time levels at different locations, because the sensors on transport public vehicles complement the readings from stationary sensors and the only ground level monitoring station in Uppsala.

SANS research group plans to deploy a new network of sensors to monitor air pollution in Barcelona. Once gathered the measurements for a period of time, they want to analyze the data sets using the GSP framework

techniques. Sadly, this is not the best time to do so, given that the COVID-19 pandemic is sweeping the entire world. Instead, this thesis is an exploratory work that will provide a first approach of these techniques, and it will show the potential that GSP has, relying in its ability to approach existing problems from different perspectives.

2.3 Applications of GSP

Almost every aspect of our life is being recorded. Up to recent time, data processing was dealing with standard or regular domains, such as time series, images in 2D, etc. Now data resides on irregular domains and complex structures that do not lend themselves to standards tools. Graphs offer the possibility to work on complicated data models, where each node has its own attributes and they are connected between weighted or non-weighted edges, modeling a wide variety of network types.

Networks are present in very different application domains, where graphs can provide a generic representation of the structure present in the data set. In this section, we consider four different scenarios, where the scale and the domain of the networks are considered very different.

We will present physical networks, both large scale (sensor networks) and human-scale ones (biological networks), where the goal is to use measurements to have a better understanding of physical phenomena.

Also, logical networks, where GSP is introduced as an alternative to existing signal processing techniques (images), or as a tool to analyze large scale sets (machine learning an data science applications).

2.3.1 Image Processing

While GSP is often applied to data sets that naturally have irregular structures, it has also been applied to other data sets where conventional signal processing has been used for many years, for example, images and video as a set of pixels, associating each pixel with a vertex, forming a regular graph where all the edges are equal to 1.

Using regular line and grid graph topology with unequal edge weights can be adapted to specific characteristics of an image or a set of images. A first approach associates different graphs to each image, using smaller edge weights to connect pixels that are on opposite sides of an image contour. Those graphs are used to capture the geometric structure in images. Such contours carry crucial visual information, in order to avoid blurring them during a filter process.

2.3.2 Biological Networks

Biological networks have proved to be a popular topic application domain for GSP. Recent research focus on the analysis of data from systems known to have a complex network architecture, such as the human brain.

Several works studied this area using the GSP framework. It has been observed that human brain activity signals can be mapped to a network (graph) where each node is a brain region. The network links (edge weights) are considered to be known *a priori* and represent the structural connectivity between brain regions [31][32], as we can see in the figure 2.1.

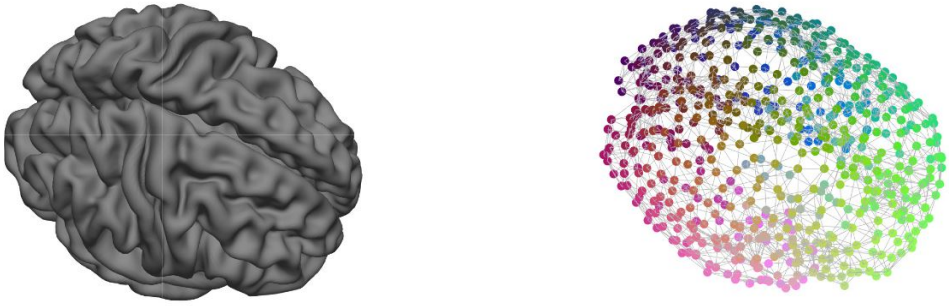


Figure 2.1: On the left side, digital human brain. On the right side, a graph representing the human brain.

GSP tools such as the graph signal representation can be used then, to analyze the brain activity signal on the structural brain network. For example, low frequencies in the graph signal represent similar activities in regions that are highly connected in the functional brain networks, while high frequencies denote very different activities in such brain regions.

It is important to denote that topics like protein interactions have been also addressed with GSP.

2.3.3 Sensor Networks

Sensor networks is one of the most natural things that comes to mind when you think about GSP. A graph represents the relative positions of sensors in the environment, and the application goals include compression, denoising, reconstruction or distributed processing of sensor data. Some initial graph-based explorations were focused on sensor networks [33][34][35].

There are different approaches to define a graph associated to a sensor network, for example, choosing edge weights as a decreasing function

of the distance between sensors. Data observations that are similar at neighboring nodes lead to a naturally smooth (low-pass) graph signals.

Such a smooth graph signal model makes it possible to detect outliers or abnormal values by high-pass filtering and thresholding, or to build effective signal reconstruction methods from sparse set of sensor readings which can potentially lead to significant savings in energy resources, bandwidth, and latency in sensor network applications. In this application cases, as we have mentioned before, GSP tools has been used to monitor air pollution, to analyze power consumption or traffic and mobility in large cities.

In [36], Ireneusz Jabloński initiates a discussion on the application of GSP to the exploration of complex and heterogeneous data systems, especially for environmental monitoring in smart habitat of city or country, using over a hundred O_3 measurements in all Poland. Ireneusz creates and performs spectral graph analysis and clustering analysis on top of the graph. In figure 2.2 we can see a graph representation of the sensor network of the region.

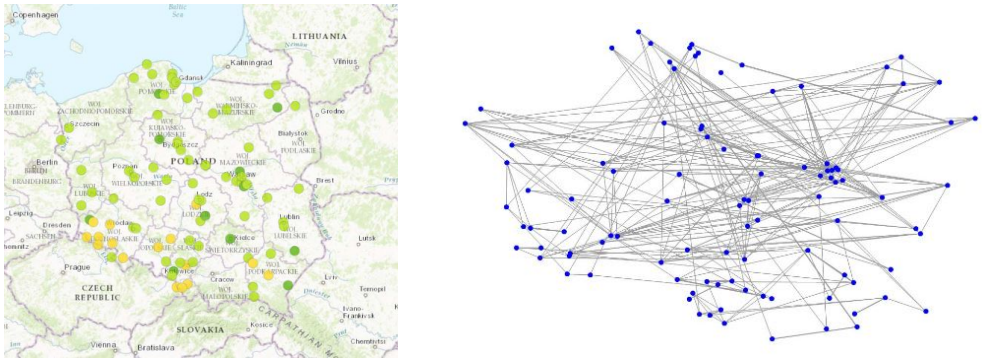


Figure 2.2: On the left side, the sensors placed over a map of Poland. On the right side, the connection between nodes.

2.3.4 Machine Learning and Data Science

Graph methods have long played an important role in machine learning applications, as they provide a natural way to represent the structure of a data set. In this context, each vertex represents one data point to which a label can be associated, and a graph can be formed by connecting vertices with edge weights that are assigned based on a decreasing function of the distance between data points in the feature space. GSP then enables different types of processing, learning, or filtering operations on values associated to graph vertices.

In a different context, GSP elements can be helpful to construct architectures to classify signals that live on irregular structures.

Graphs can be constructed to describe similarities between users or items in recommendation systems that assists customers in making decisions by collecting information about how other users rate particular services or items [37].

Content based recommendation can also be addressed as an online learning problem, where the key idea is to represent the reward function in an online recommendation system as a linear combination of eigenvectors of the similiarity graph that connects different items. Using this representation, it is possible to optimize the reward function by favoring smoothness on the graph, which is effective in video recommendation examples [38].

These examples provide evidence for the potential benefits of using GSP principles in big data applications.

Finally, GSP framework can also be used to design architectures to analyze or classify whole graph signals that originally live on irregular structures. In particular, the GSP toolbox has been extensively used to extend convolutional deep learning techniques to data defined on graphs. For example, the convolutional neural network paradigm has been generalized with the help of GSP elements for the extraction of feature descriptors for 3D shapes [39][40].

These examples provide evidence for the potential benefits of using GSP principles in big data applications.

Chapter 3

GSP Theory

This section of the document aims to give more mathematical knowledge of the basis of GSP. Starting with how the initial graph is created, then revising the concepts of adjacency and Laplacian matrices, going through the computation of the Graph Discrete Fourier Transform (GDFT) and its inverse (IGDFT), and finishing with the concept of filter and the methods used on the reconstruction of the signal.

3.1 Graph Creation

For a graph that corresponds to a network with geometrically distributed vertices, it is natural to relate the edge weights with the distance between vertices. Considering m and n as a vertices whose location in space are defined by the coordinates r_m and r_n, we have that the Euclidean distance between these two vertices is then:

$$r_{mn} = distance(m, n) = \|r_m - r_n\|_2 \tag{3.1}$$

A common way to define the graph weights in such networks is through an exponentially decaying function of the distance, such as

$$W_{mn} = \begin{cases} e^{-r_{mn}^2/\tau^2}, & \text{for } r_{mn} \leq k \\ 0, & \text{for } r_{mn} > k \text{ and } m = n \end{cases} \tag{3.2}$$

where $\mathrm{r_{mn}}$ is the Euclidean distance between the vertices m and n, and τ and k are chosen constants. Different values will be explored in chapter *Evaluation and Results* while using this function. This is also physically

well justified, as based on $e^{-r_{mn}^2/\tau^2}$ the weights tend to 1 for closely spaced vertices and diminish for distant vertices.

The rationale for this definition of edge weights is the assumption that the signal value measured at a vertex n is similar to signal values measured at its neighboring vertices. Then, the estimation of a signal at a vertex n should also involve neighboring vertices connected with large weights (close to 1), while the signal values at farther distances would be less relevant, with lower weight coefficients or even not included at all.

The Gaussian function used in 3.2, is appropriate in many applications, however, there are many other forms to penalize data values associated with the vertices which are far from the considered vertex, such as

$$W_{mn} = \begin{cases} e^{-r_{mn}/\tau}, & \text{for } r_{mn} \leq k \\ 0, & \text{for } r_{mn} > k \text{ and } m = n \end{cases} \tag{3.3}$$

or the inverse Euclidean distance between vertices m and n, given by

$$W_{mn} = \begin{cases} \frac{1}{r_{mn}}, & \text{for } r_{mn} \leq k \\ 0, & \text{for } r_{mn} > k \text{ and } m = n \end{cases} \tag{3.4}$$

All the elements of the weighted matrix are greater or equal than zero.

3.2 Adjacency and Laplacian Matrix

For a given set of vertices and edges, a graph can also be formally represented by its adjacency matrix A, which describes the vertex connectivity.

For N distinct vertices, A is an $N \times N$ matrix, where each element A_{mn} of the adjacency matrix A assumes values $A_{mn} \in 0, 1$. The value $A_{mn} = 0$ is assigned if the vertices m and n are not connected with and edge, and $A_{mn} = 1$ if these vertices are connected, that is

$$A_{mn} \overset{\text{def}}{=} \begin{cases} 1, & \text{if there is a connection} \\ 0, & \text{otherwise} \end{cases} \tag{3.5}$$

Adjacency matrices fully reflect the structure arising from the topology of data acquisition, where a non-symmetric matrix represents a directed graph and a symmetric one represents a undirected graph.

We now introduce the concept of degree matrix, D, which, for an undirected graph, is a diagonal matrix with elements D_{mm}, equal to the sum of weights of all edges connected to the vertex m, that is, the sum of elements in its m-th row

$$D_{mm} \stackrel{\text{def}}{=} \sum_{n=0}^{N-1} W_{mm}. \tag{3.6}$$

Another important descriptor of graph connectivity is the graph Laplacian Matrix, L, which combines the weight matrix and the degree matrix. It is defined as follows

$$L \stackrel{\text{def}}{=} D - W, \tag{3.7}$$

where W is the weighted matrix and D the diagonal degree matrix with elements $D_{mm} = \sum_n W_{mm}$. The elements of a Laplacian matrix are nonnegative real numbers at the diagonal positions and non positive real numbers at the off-diagonal positions. For an undirected graph, the Laplacian matrix is symmetric, $L = L^T$.

For practical reasons, it is often advantageous to use the normalized Laplacian, defined as

$$L \stackrel{\text{def}}{=} D^{-1/2}(D - W)D^{-1/2} = I - D^{-1/2}WD^{-1/2}, \tag{3.8}$$

3.3 GDFT and IGDFT

Classical exploratory data analysis often employs estimation of signals in the spectral (Fourier) domain; this has led to a number of simple and efficient algorithms, while standard spectral analysis employs an equidistant grid in both time and frequency.

Following the ideas of a system in a graph, we next show that spectral domain representation of graph signals are naturally based on spectral decomposition of the adjacency matrix or graph Laplacian.

The graph Fourier transform (GFT) of a signal x, is defined as

$$X = U^{-1}x, \tag{3.9}$$

where X denotes a vector of the GDFT coefficients, and U is a matrix whose columns represent the eigenvectors of the adjacency matrix,

A. The elements of X are denoted by $X(k)$, for k = 0, 1, ..., N-1, and because the adjacency matrix is symmetric for undirected graphs, the eigenmatrices of a symmetric matrix satisfy the property

$$U^{-1} = U^T. \tag{3.10}$$

The element, $X(k)$, of the GFT vector X, therefore represents a projection of the considered graph signal, $x(n)$, on to the k-eigenvector of A, given by

$$X(k) = \sum_{n=0}^{N-1} x(n)u_k(n). \tag{3.11}$$

In this way, the GDFT can be interpreted as a set of projections (signal decomposition) onto the set of eigenvectors, $u_0, u_1, ..., u_{N-1}$.

The IGDFT is then straightforwardly obtained from 3.9 as

$$x = UX, \tag{3.12}$$

or element-wise

$$x(n) = \sum_{k=0}^{N-1} X(k)u_k(n). \tag{3.13}$$

3.4 Filters

In signal processing, a filter is a process that removes unwanted components or features from a signal. Filtering is a class of signal processing.

Most often, this means removing some frequencies or frequency bands. However, filters do not exclusively act in the frequency domain; especially in the field of image processing, many other targets for filtering exists.

Filters are widely used in electronics and telecommunications, in radio, television, audio recording, image processing, etc.

There are many different bases of classifying filters and these overlap in many ways, meaning that there is no simple hierarchical classification. Here we have a list of some filters:

- non-linear or linear (e.g. low-pass filters, high-pass filters)

- time-variant or time-invariant, also known as shift-invariance
- analog or digital
- infinite impulse response (IIR) or finite impulse response (FIR)

3.5 Methods for Signal Reconstruction

As I have mentioned in *Goals* we want to explore prediction techniques to predict sensor values in specific points of the network. In this section, a quick explanation of those techniques will be made.

3.5.1 Linear Combination using Laplacian Coefficients

This first method reconstructs the signal on a station using neighboring nodes and the values from the Laplacian matrix. We have that y_n is the vector which will contain the reconstructed values. For example, in station $m = 3$ has 3 neighboring stations, $n = \{1, 5, 7\}$, then we can obtain those values using the following

$$y_n = a_1 x_1 + a_5 x_5 + a_7 x_7. \tag{3.14}$$

Where a_m are the coefficients of the Laplacian matrix that connect the n nodes with m.

3.5.2 Multiple Linear Regression

The second method is practically the same equation, but the reconstruction is based on Machine Learning.

$$y_n = a_0 + a_1 x_1 + a_5 x_5 + a_7 x_7. \tag{3.15}$$

Here, we need to calculate the offset a_0 (if exists). Then we train the multiple linear regression model with a subset of the data set.

3.5.3 GSP

In this method we consider that the graph signal is of a low-pass nature. Such a signal can be expressed as a linear combination of $K < N$

eigenvectors of the graph Laplacian which exhibit the lowest smoothness indices,

$$x(n) = \sum_{k=0}^{K-1} X(k)u_k(n), n = 0, 1, ..., N-1. \tag{3.16}$$

The GDFT domain coefficients of this (K-sparse) signal in the GDFT domain are of the following form

$$X = [X(0), X(1), ..., X(K-1), 0, 0, ..., 0]^T. \tag{3.17}$$

Recall that a graph signal is sparse in the GDFT domain if $K \ll N$. The smallest number of graph signal samples, M, needed to recover the sparse signal is therefore $M = K < N$. For stability of reconstruction, it is common to employ $K \leq M < N$ graph signal samples. The vector of available graph signal samples will be referred as the *measurement vector*, and will be denoted by y, while the set of vertices over which the sample of graph signals are available is denoted by

$$\mathbb{M} = \{n_1, n_2, ..., n_M\} \tag{3.18}$$

Then, the measurement matrix can now be defined using the IGDFT, $x = UX$, of which an element-wise form is given by 3.16. This equation corresponding to the available graph singal samples at vertices $n \in M = \{n_1, n_2, ..., n_M\}$ then define the system

$$\begin{bmatrix} x(n_1) \\ x(n_2) \\ \vdots \\ x(n_M) \end{bmatrix} = \begin{bmatrix} u_0(n_1) & u_1(n_1) & \dots & u_{N-1}(n_1) \\ u_0(n_2) & u_1(n_2) & \dots & u_{N-1}(n_2) \\ \vdots & \vdots & \ddots & \vdots \\ u_0(n_M) & u_1(n_M) & \dots & u_{N-1}(n_M) \end{bmatrix} \begin{bmatrix} X(0) \\ X(1) \\ \vdots \\ X(N-1) \end{bmatrix} \tag{3.19}$$

for which the matrix form is given by

$$y = A_{MN}X, \tag{3.20}$$

where A_{MN} is the *measurement matrix* and the *measurements vector*

$$y = [x(n_1), x(n_2), \dots, x(n_M)]^T \tag{3.21}$$

consists of the available graph signal samples. In general, since $M < N$ this system is undetermined, and cannot be solved for X without additional constrains.

The assumption that the spectral representation of a signal contains a linear combination of only $K \leq M$ slowest varying eigenvectors allows us to exclude the GDFT coefficients $X(K), X(K+1), \ldots, X(N-1)$ in 3.17, since these are zero-valued and do not contribute. With this in mind, the MxN system in 3.20 is reduced to the following MxK system

$$\begin{bmatrix} x(n_1) \\ x(n_2) \\ \vdots \\ x(n_M) \end{bmatrix} = \begin{bmatrix} u_0(n_1) & u_1(n_1) & \ldots & u_{K-1}(n_1) \\ u_0(n_2) & u_1(n_2) & \ldots & u_{K-1}(n_2) \\ \vdots & \vdots & \ddots & \vdots \\ u_0(n_M) & u_1(n_M) & \ldots & u_{K-1}(n_M) \end{bmatrix} \begin{bmatrix} X(0) \\ X(1) \\ \vdots \\ X(K-1) \end{bmatrix} \tag{3.22}$$

or in the matrix form

$$y = A_{MK} X_K \tag{3.23}$$

where the definitions of the reduced measurement matrix A_{MK} and the reduced GDFT vectors X_K are obvious. For $M = K$ independent measurements, this system can be solved uniquely, while for $M > K$ is typically overdetermined and the solution is found as

$$X_K = (A_{MK}^T A_{MK})^{-1} A_{MK}^T y = pinv(A_{MK}) y, \tag{3.24}$$

where $pinv(A_{MK}) = (A_{MK}^T A_{MK})^{-1} A_{MK}^T$ is the matrix pseudo-inverse of A_{MK}.

After X_K is calculated, all GDFT values follow directly as 3.17. The graph signal is then recovered at all vertices using $x = UX$.

3.5.4 Comparing methods

It is important to notice that the first method is very dependant on the Laplacian values, as is based on the distances between nodes. Both graph and coefficients depend on how we build the Laplacian Matrix.

In both MLR and GSP only the graph depends on the distances, but not the coefficients to reconstruct the signal.

Specially in MLR, the coefficients are optimal in terms of linearity and the neighbor nodes are chosen according to the graph but it is impossible to use if the data set has gaps.

On the other hand, the GSP technique is a filter, and as such, it considers a low-pass signal and it tries to make the signal smooth, i.e. with few sudden jumps with respect the neighbors, and although GSP is not optimal in the lineal sense, it has the advantage that it can be used with data gaps.

During this part, we evaluate the results using different values of k and distances between stations. To check the performance of the previous methods, RMSE and R2 will be used to compare the predicted values y_n against the real ones.

Chapter 4

Data set

The data sets for this Master Thesis are extracted from the website *Transparència de Catalunya* [41], managed by the *Generalitat de Catalunya*.

It is an online system that allows people a free and easy access to open information regarding local administrations; in this case, data from the air pollution.

Data (once refined) is arranged in a table, per each month and each pollutant, using the following fields as a columns

- Code EOI: It is a code that is used to identify the station.
- Nom Estació: Name of the station.
- Municipi: Location of the station.
- Latitud: Latitude.
- Longitud: Longitude.
- Altitud: Altitude over the sea level.
- Tipus Estació: Type of station.
- Data: Date of the measurements. It uses the DD/MM/YY format.
- H01, ..., H24: It goes from column H01 to Column H24 and represents the time of the day when the samples are retrieved, from 01:00:00 to 00:00:00.

We have data regarding the month of January, 2020, from 01-01-2020 to 31-01-2020. Such data is gathered by the sensors once each hour. The

table below shows a summary of the amount of cleaned samples and the total number of stations, per pollutant and per month.

Month	Pollutant	#Stations	#Samples
January	O_3	16	610
	NO_2	28	455
	PM_10	14	402

Table 4.1: Table that summarizes the amount of data used.

As we can see in the previous table, we don't have the same amount of data for each pollutant. That is because the stations produce gaps in the data due to maintenance, sensor errors, electrical shortage during storms, etc; This implies that we won't have information for all the possible timestamps in a month.

Pollutant	Units
O3	$\mu g/m^3$
NO2	$\mu g/m^3$
PM10	$\mu g/m^3$

Table 4.2: Units of measurement

To be able to see weather the results of the RMSE indicates a big error on the prediction, here we have the Mean Values (MV) and the Standard Deviation (STD) per pollutant during the month of January.

Station	MV	STD
Badalona	21.96	22.20
Barcelona (Eixample)	18.87	18.75
Barcelona (Gràcia - Sant Gervasi)	25.96	20.26
Barcelona (Ciutadella)	21.91	22.18
Barcelona (Parc Vall Hebron)	35.55	21.44
Barcelona (Palau Reial)	34.92	22.79
Barcelona (Observatori Fabra)	63.61	13.96
Gavà	33.91	23.02
Montcada i Reixac	14.81	21.05
El Prat de Llobregat (Sagnier)	22.07	23.83
Rubí	22.3	22.44
Sabadell	19.39	18.25
Sant Adrià de Besòs	18.89	23.05
Sant Cugat del Vallès	16.83	21.38
Sant Vicenç dels Horts (Ribot)	15.10	21.29
Viladecans - Atrium	31.13	23.177

Table 4.3: Table of means and standard deviations for O_3.

Station	MV	STD
Badalona	37.11	20.19
Barcelona (Poblenou)	40.18	20.68
Barcelona (Sants)	36.70	20.01
Barcelona (Eixample)	49.82	22.05
Barcelona (Gràcia - Sant Gervasi)	45.65	22.86
Barcelona (Ciutadella)	37.22	19.15
Barcelona (Palau Reial)	30.88	20.11
Barcelona (Observatori Fabra)	9.09	8.57
Gavà	18.50	9.79
L'Hospitalet de Llobregat	40.04	20.96
Martorell	28.92	14.18
Mollet del Vallès	39.46	20.55
Montcada i Reixac	33.36	18.41
Pallejà (Roca de Vilana)	20.82	11.66
El Prat de Llobregat (Jardins de la Pau)	33.44	18.93
El Prat de Llobregat (Sagnier)	34.39	17.91
Rubí	26.43	18.48
Sabadell	32.10	19.78
Sant Adrià del Besòs	39.11	22.41
Sant Andreu de la Barca	35.12	16.04
Sant Cugat del Vallès	29.31	15.68
Santa Coloma de Gramanet	36.65	19.31
Barberà del Vallès	35.74	20.67
Santa Perpètua de Mogoda	32.02	18.03
Sant Vicenç dels Horts (Ribot)	32.11	15.89
Sant Vicenç dels Horts	33.16	15.61
Viladecans - Altrium	27.78	15.44

Table 4.4: Table of means and standard deviation for NO_2.

Station	MV	STD
Barcelona (Poblenou)	24.58	14.85
Barcelona (Eixample)	23.52	14.02
Barcelona (Gràcia - Sant Gervasi)	21.37	13.15
Barcelona (Parc Vall Hebron)	14.30	10.51
Barcelona (Palau Reial)	18.43	10.51
Barcelona (Observatori Fabra)	9.95	5.65
L'Hospitalet de Llobregat	18.79	12.11
Montcada i Reixac	22.54	11.36
Montcada i Reixac (Can Sant Joan)	21.01	11.22
Rubí	16.74	10.07
Sabadell	25.05	15.35
Sant Adrià del Besòs	23.23	12.97
Santa Perpètua de Mogoda	23.71	12.43
Sant Vicenç dels Horts (Ribot)	25.20	14.09

Table 4.5: Table of means and standard deviation for PM_{10}.

Chapter 5

Evaluation and Results

To do this evaluation, I have used PyGSP [42]. This library facilitates a wide variety of operations on graphs, like computing their Fourier basis. Its core is spectral graph theory, and many of the provided operations scale to very large graphs. The chapter is organized as follows

- Section 5.1 presents the different scenarios that will be evaluated. Distances between stations, edges and location of the stations on a map and the level of induced Gaussian noise with $\mu = 0$ and $\sigma =$ (mean value of a node * percentage of error) to a station, transforming that particular station in a faulty one.
- Section 5.2 onward presents the distribution of nodes, and results of using the above-mentioned methods for each of the pollutants. RMSE and R2 will show the performance of procedures.

5.1 Scenarios

These are the variables that we have in consideration:

- Maximum Distance Between Stations = 10km, 15km or 20km.
- Value of K (GSP):
  - 1, 4 or 8, for O_3 and PM_{10}.
  - 1, 4, 8 or 12, for NO_2. [1]
- Gaussian noise with percentage of error: 10, 15, 20%.

[1]For stability of reconstruction, it is commonly employed $K \leq M < N$ graph signal samples. We avoid the value 12 for O_3 and PM_{10} as it is close to M. See 3.18.

5.2 Pollutant: O_3

This section is centered in the O_3 pollutant. It presents the selected stations, the adjacencies between nodes, and the performance of the evaluation methods.

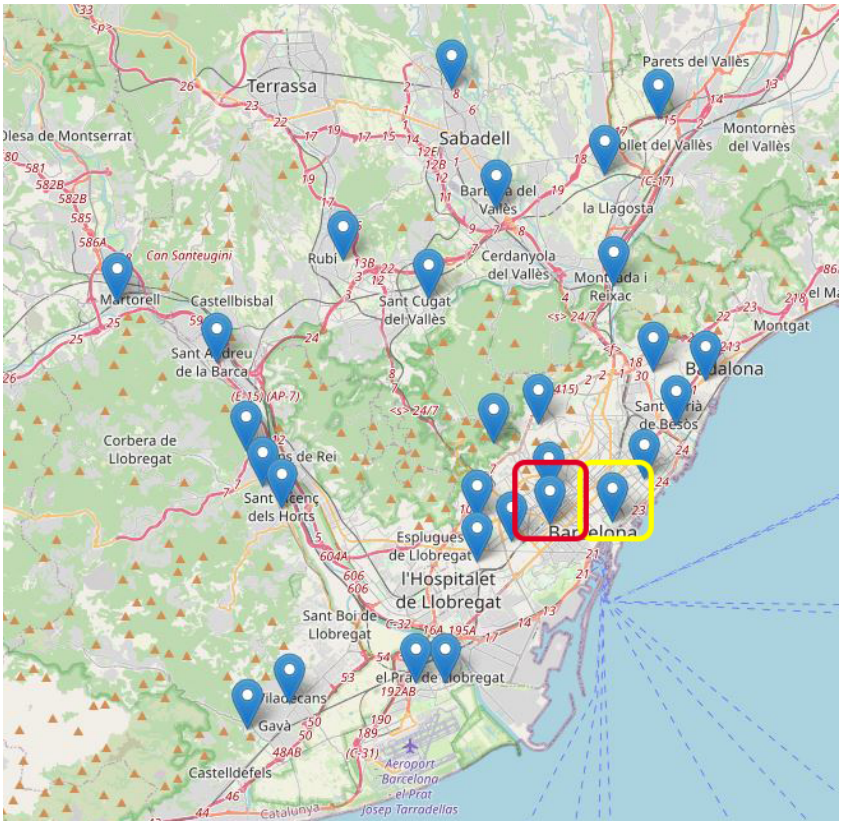
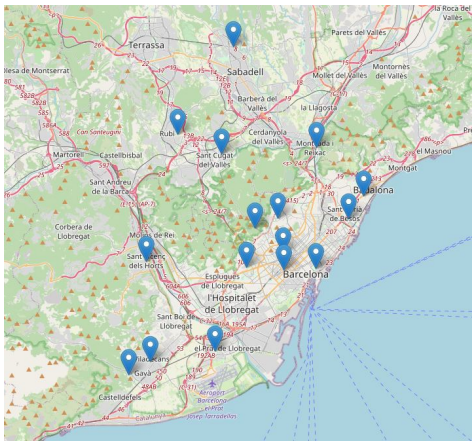
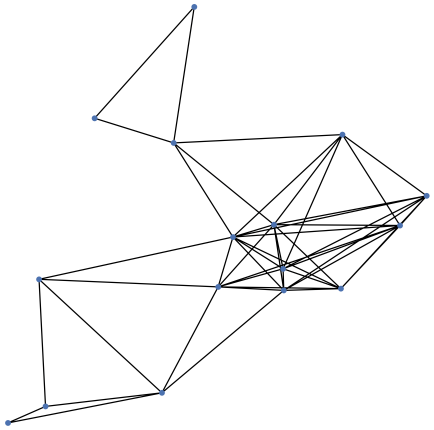


Figure 5.1: In red, the station to reconstruct the signal: Barcelona (Eixample). In yellow, the faulty station: Barcelona (Ciutadella).

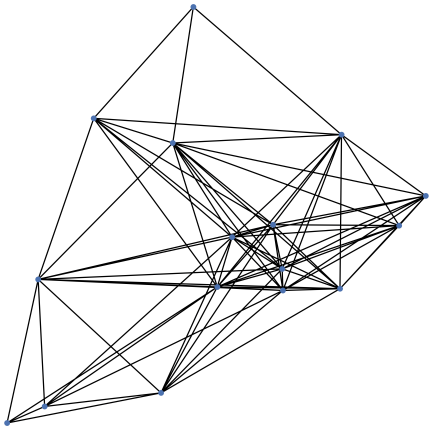
Figure 5.2 show the connections between nodes when using different distances, for values of 10, 15 and 20 Km. It is easy to see that as the distance increases, the evaluation methods that depend on the graph adjacencies will have more samples to use in the model, and theoretically, a better performance.



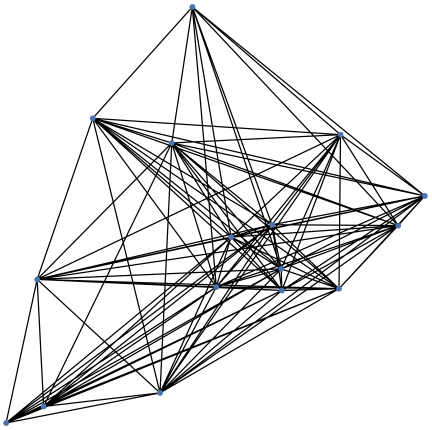
(a) O_3 sensor nodes in Barcelona.



(b) Max distance = 10 Km.



(c) Max distance = 15 Km.



(d) Max Distance = 20 Km.

Figure 5.2: Different cases of maximum distance between stations that measure O_3.

The following images contain different cases. Each case shows the RMSE/R2 performance of an execution of LC, MLR and GSP methods with different parameters. Notice that the case denoted with 0% implies that there is no faulty station on that particular execution.

5.2.1 RMSE

In figure 5.3 we can see that MLR has the best performance of all three methods for each of the thresholds. LC and GSP are much less optimal because the former, base its reconstruction only on the adjacencies of the

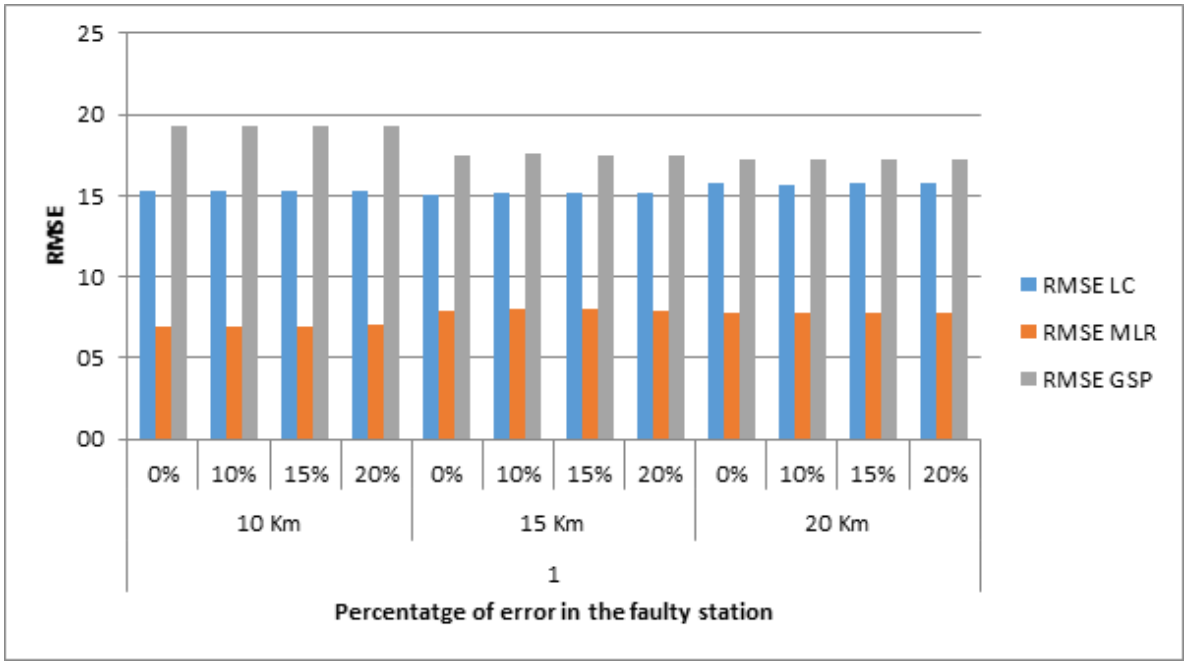


Figure 5.3: Comparison of RMSE for threshold 10, 15, 20 Km with K = 1.

target node, and the latter doesn't benefit from the correlations between nodes.

As the threshold increases, GSP improves. This method benefits from the fact that, the higher the threshold, the more nodes are included and used in the reconstruction.

Regarding the cases with faulty data, all three methods have worse performance as the threshold increases, meaning that faulty stations affects directly all methods.

In figure 5.4, in comparison with the previous iteration, we can see a high improvement on GSP, even with faulty data. That is, as the values of K increases, the amount of nodes used for reconstructing the signal is also bigger, giving a better performance.

For the cases where we have a faulty station, the performance of all the linear methods stay pretty much the same as the previous iteration.

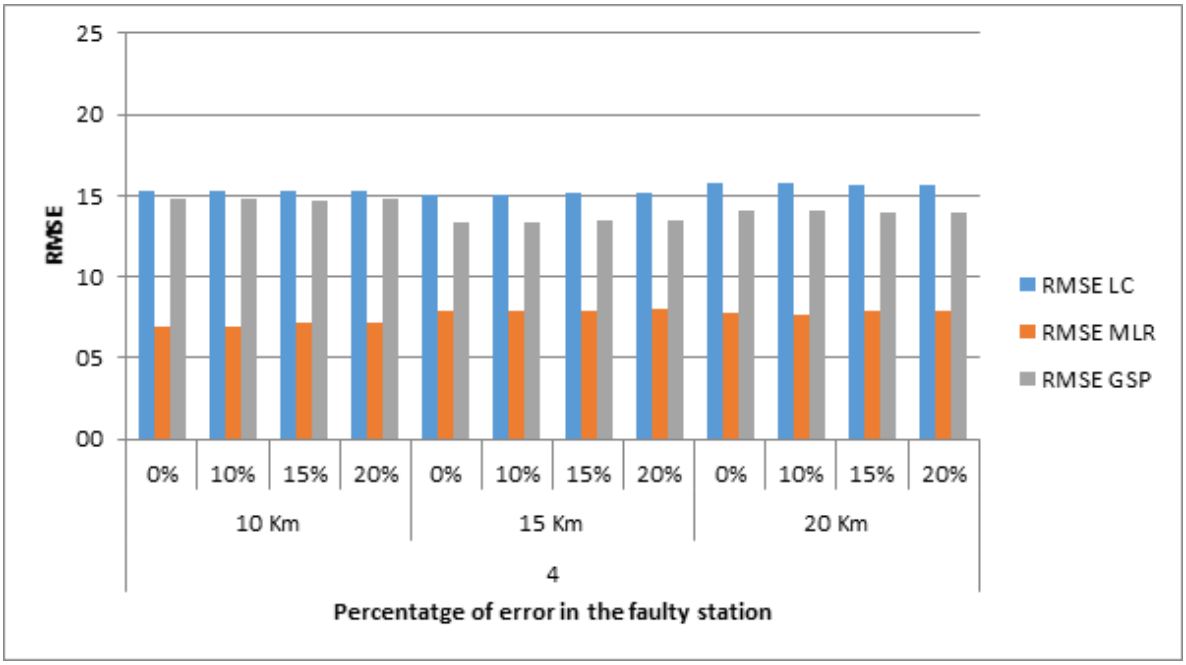


Figure 5.4: Comparison of RMSE for threshold 10, 15, 20 Km with K = 4.

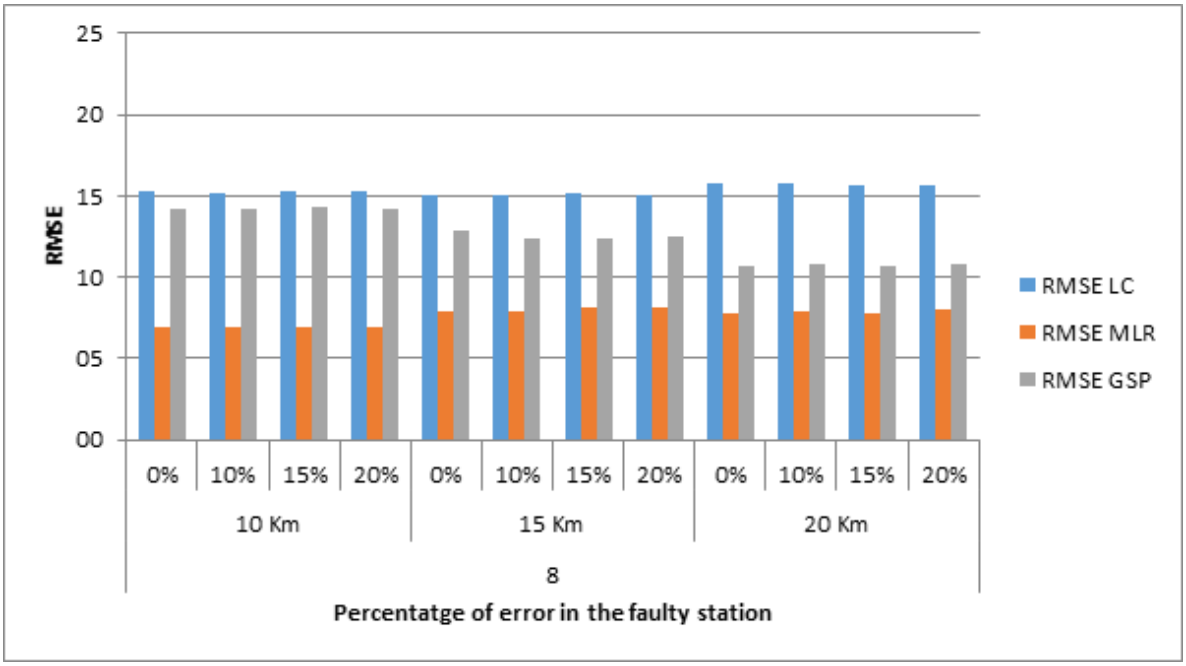


Figure 5.5: Comparison of RMSE for threshold 10, 15, 20 Km with K = 8.

In figure 5.5 the improvement of GSP for 10 and 15 Km with K = 8 is minimal. That means that there is not much difference on the performance on the first two thresholds, possibly because that 5 Km of difference doesn't include nodes that influences that much the reconstruction method for this specifid signal; but in the third one there is a notorious increase, which is a step forward towards the performance of MLR. Here, we use a value of K close to M, which means that GSP takes into account almost all the nodes.

Still, even with faulty data GSP shows a continuous improvement as the distance threshold and K increases.

The O_3 is a pollutant that is spread out all over the city and the outskirts, and it is not focused on the area around certain nodes. This match with the results we have got so far, where we have indeed a low-pass signal and GSP performs good on this kind of signals.

It is also interesting to mention that O_3 has different values depending on the season. During winter we will have lesser values than in summer.

5.2.2 R2

In figures 5.6, 5.7 and 5.8 basically we can see how well the three methods under study perform. There is not much to say about it, but we see that GSP obtains a performances close to MLR as the threshold and K increases.

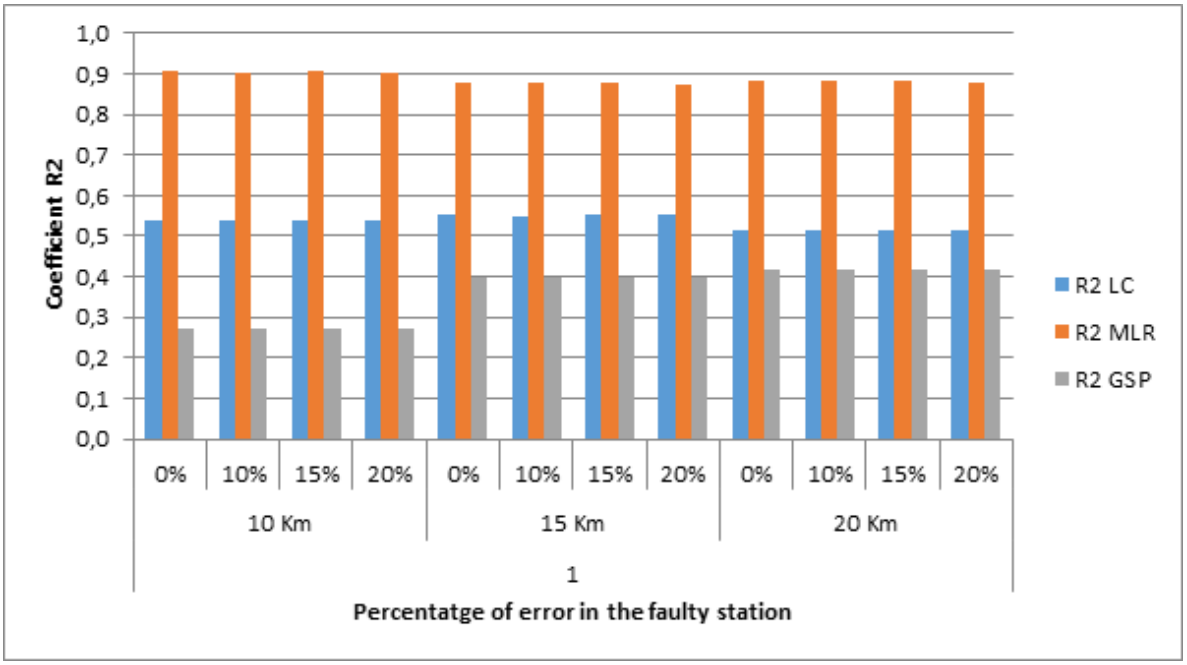


Figure 5.6: Comparison of R2 for threshold 10, 15, 20 Km with K = 1

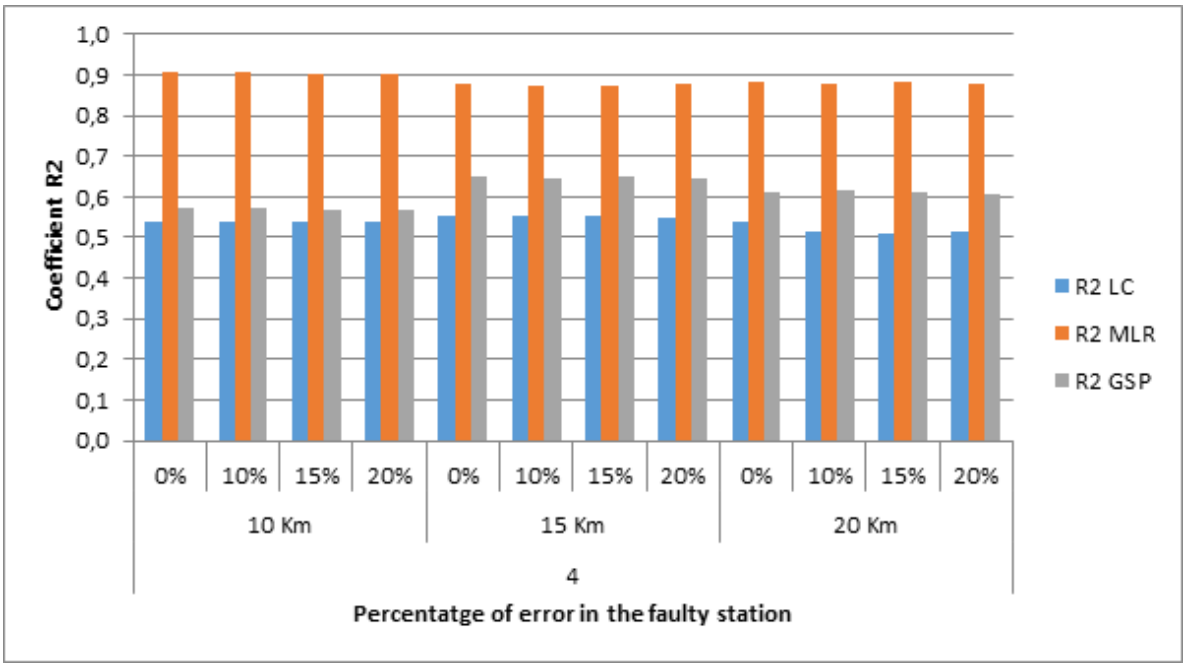


Figure 5.7: Comparison of R2 for threshold 10, 15, 20 Km with K = 4.

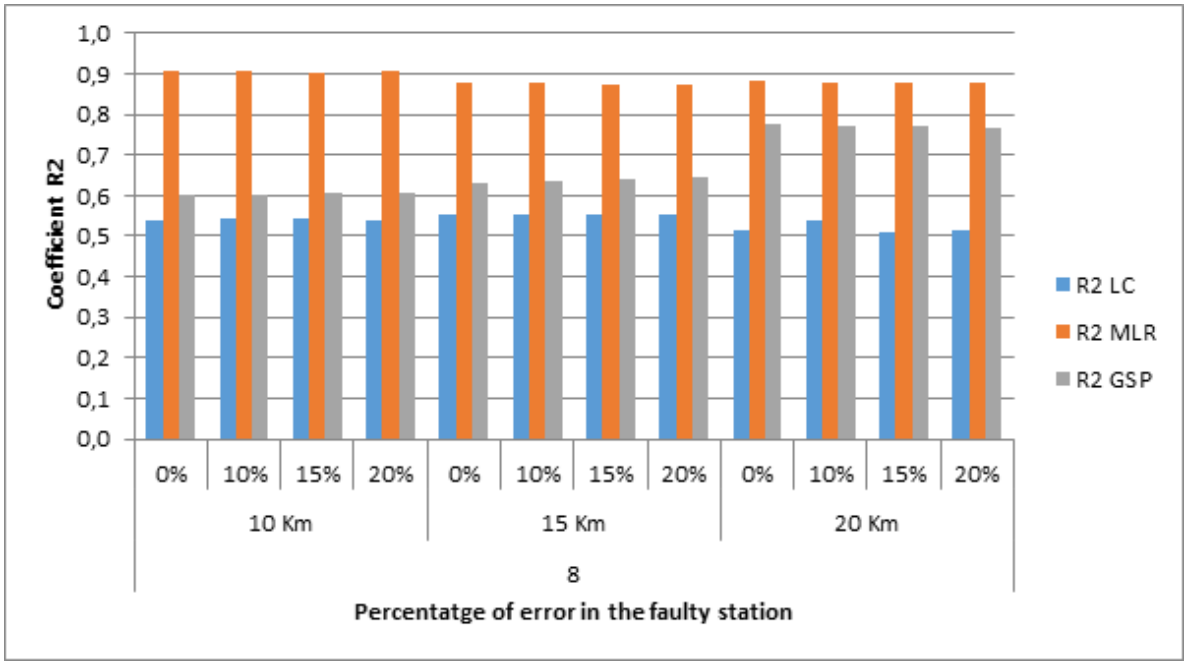


Figure 5.8: Comparison of R2 for threshold 10, 15, 20 Km with K = 8.

5.3 Pollutant NO_2

This section is centered in the NO_2 pollutant. Similarly as the previous section, here we present the selected stations, the adjacencies between nodes, and the perfomance of the evaluation methods.

Figure 5.10 show the connections between nodes when using different thresholds, for value sof 10, 15 and 20 Km.

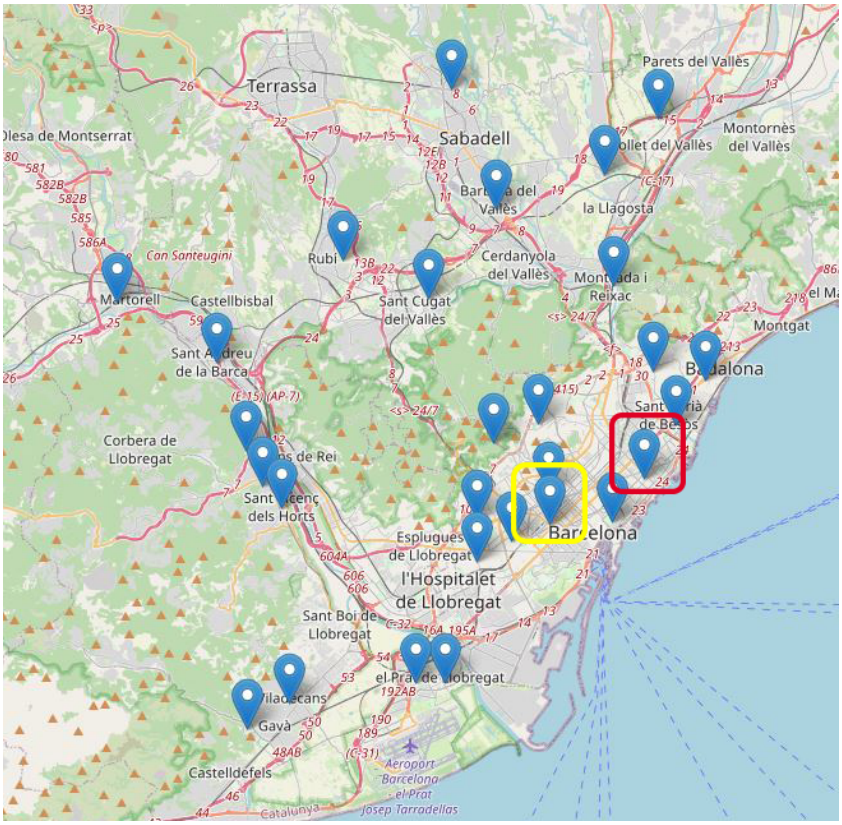


Figure 5.9: In red, the station to reconstruct the signal: Barcelona (Poblenou). In yellow, the faulty station: Barcelona (Eixample).

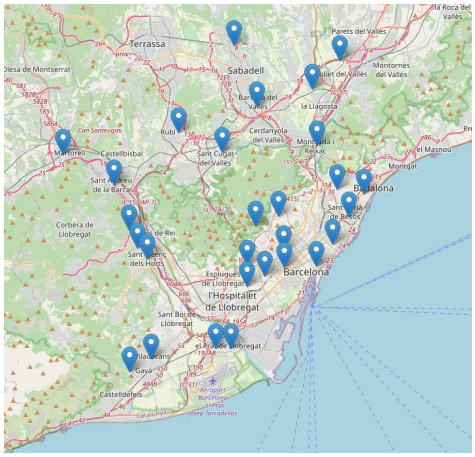
5.3.1 RMSE

In figure 5.11 we can see a more paired results. MLR is still has the best performance in all three cases, but GSP is closer to LC and MLR than it was with O_3. This could imply that the signal measured by the stations could be smoother than the previous pollutant and thus, the error obtained for all three methods is smaller.

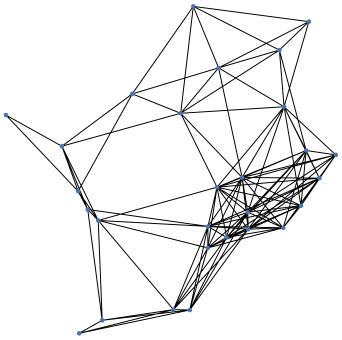
For K = 4, as shown in figure 5.12, the performance of GSP is closer to MLR. When the maximum distance threshold is used, we are able to see that GPS already surpasses LC.

In figure 5.13 the framework under study, GSP, obtains a performance very close to MLR. Because it considers both data and the graph topology, makes sense that, as the value of K increases, GSP will obtain better performance than the MLR.

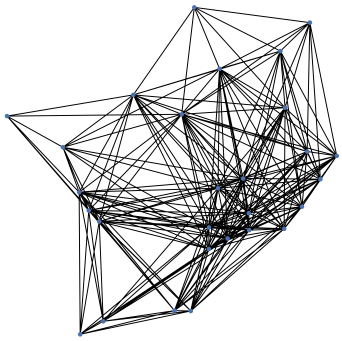
The hypothesis that NO_2 pollutant is also a low-pass signal is correct. In figure 5.14, and comparing that to previous iterations, it is clear that



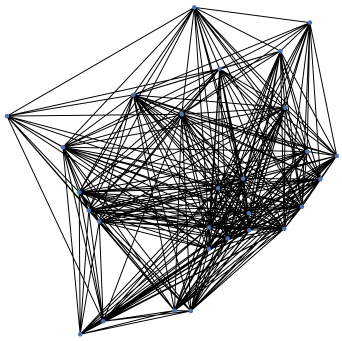
(a) NO_2 sensor nodes in Barcelona.



(b) Max distance = 10 Km.



(c) Max distance = 15 Km.



(d) Max Distance = 20 Km.

Figure 5.10: Different cases of maximum distance between stations that measure NO_2.

as the value of K increases, it affects positively GSP. LC remains in the same values for all the iterations.

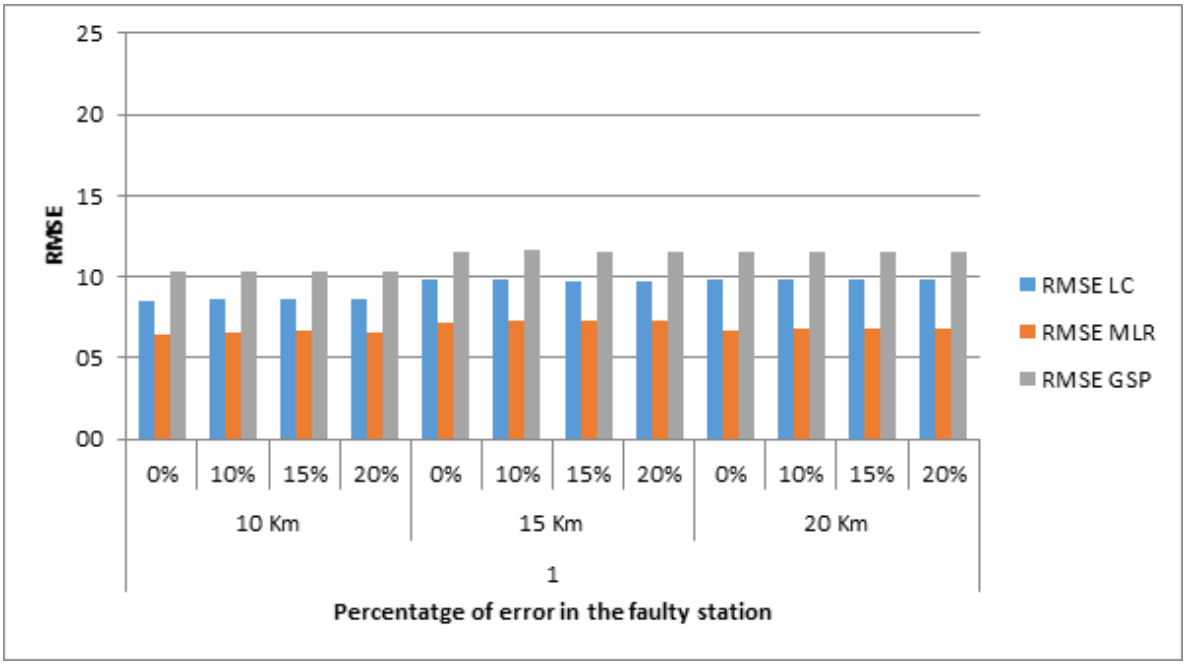


Figure 5.11: Comparison of RMSE for threshold 10, 15, 20 Km with K = 1.

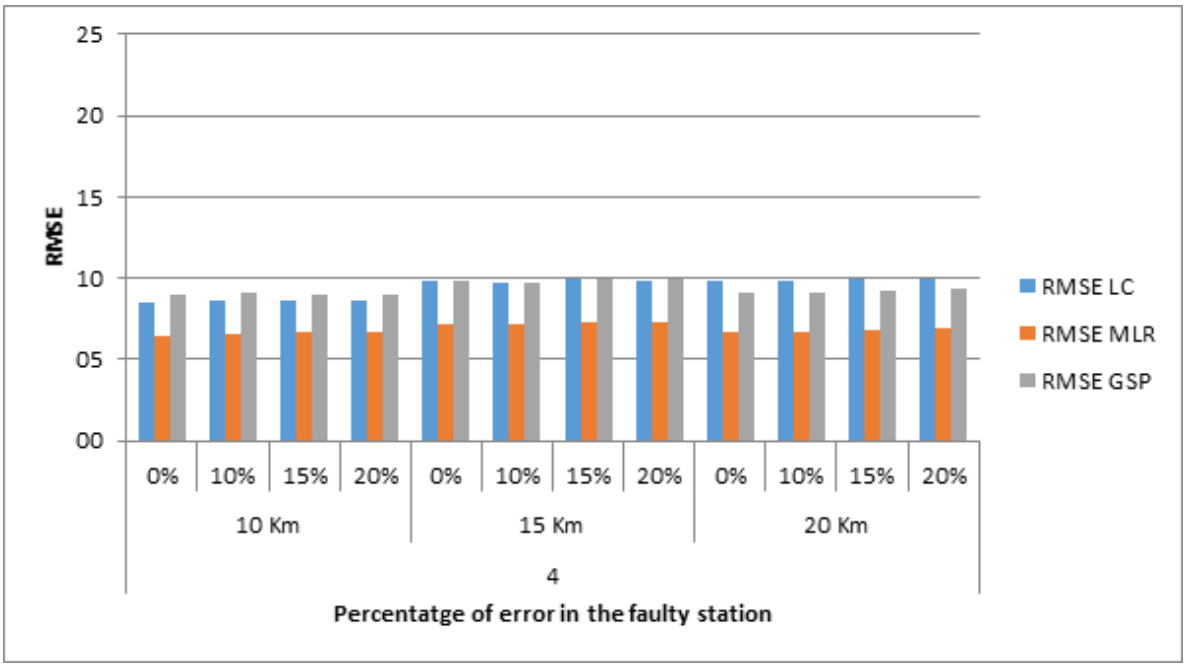


Figure 5.12: Comparison of RMSE for threshold 10, 15, 20 Km with K = 4.

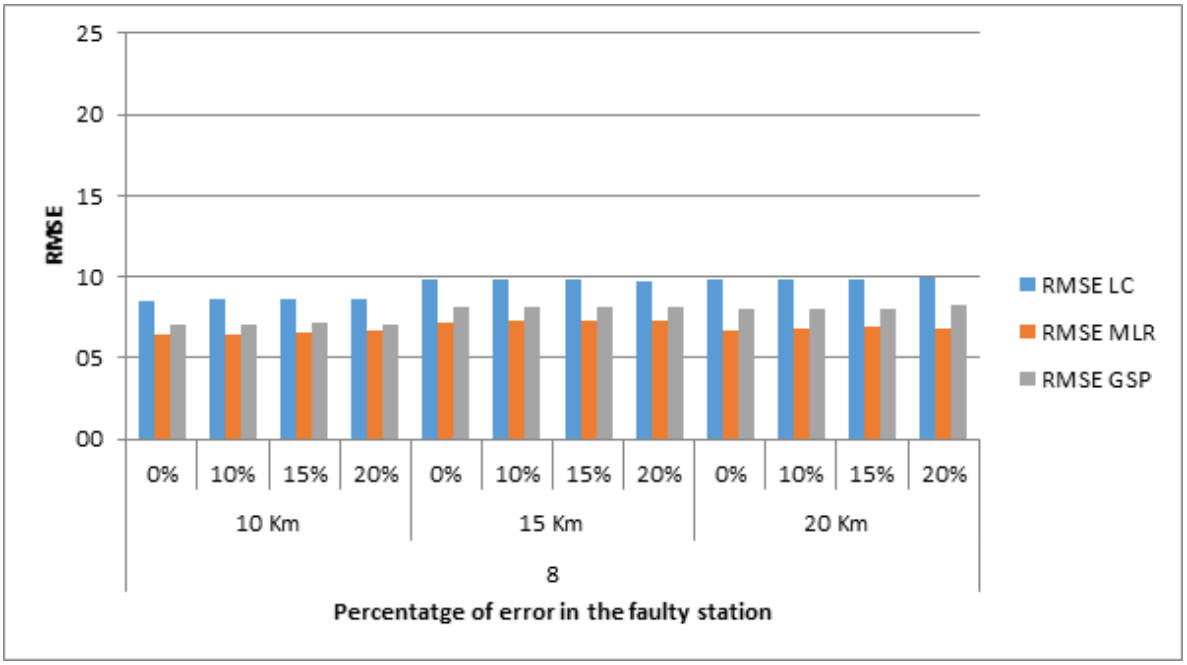


Figure 5.13: Comparison of RMSE for threshold 10, 15, 20 Km with K = 8.

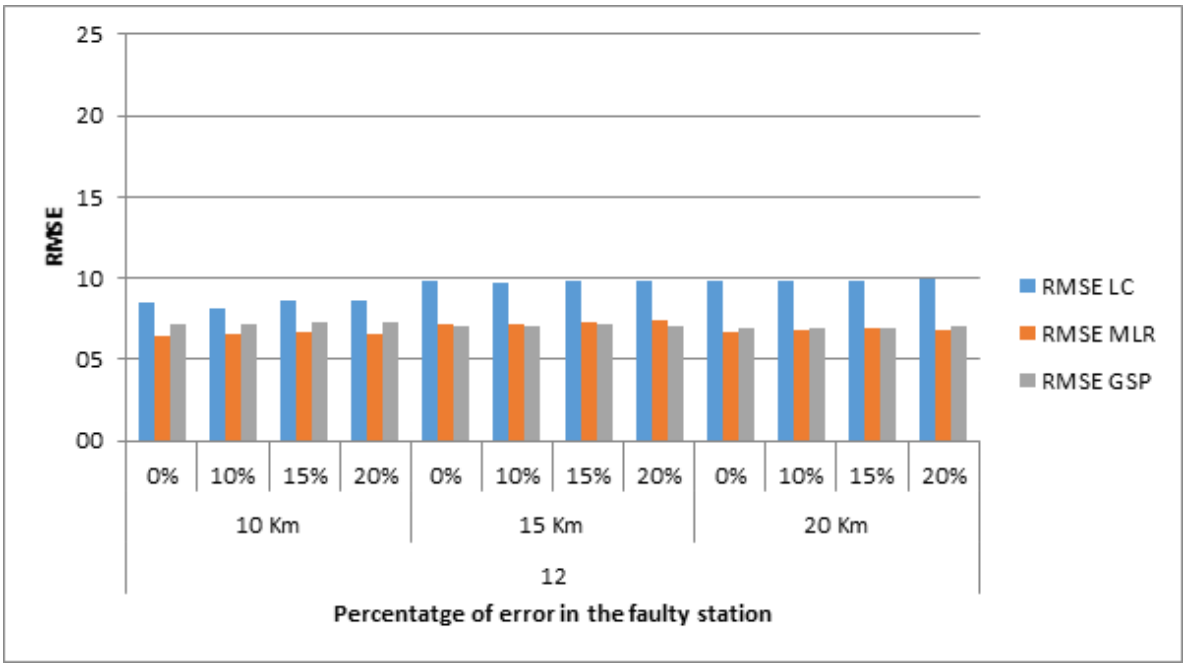


Figure 5.14: Comparison of RMSE for threshold 10, 15, 20 Km with K = 12.

5.3.2 R2

Same analysis as we have done with O_3; when the value of K increases towards M (the number of sample signals of the graph) we obtain the best performances. Here, GSP matches the score of MLR, even sometimes is better.

It seems that the NO_2 in the air it is equally distributed over the metropolitan area of Barcelona, where the stations are able to capture and build a signal which is considered a low-pass signal, being smoother than O_3.

This pollutant is not seasonal and it depends on the industries and vehicles.

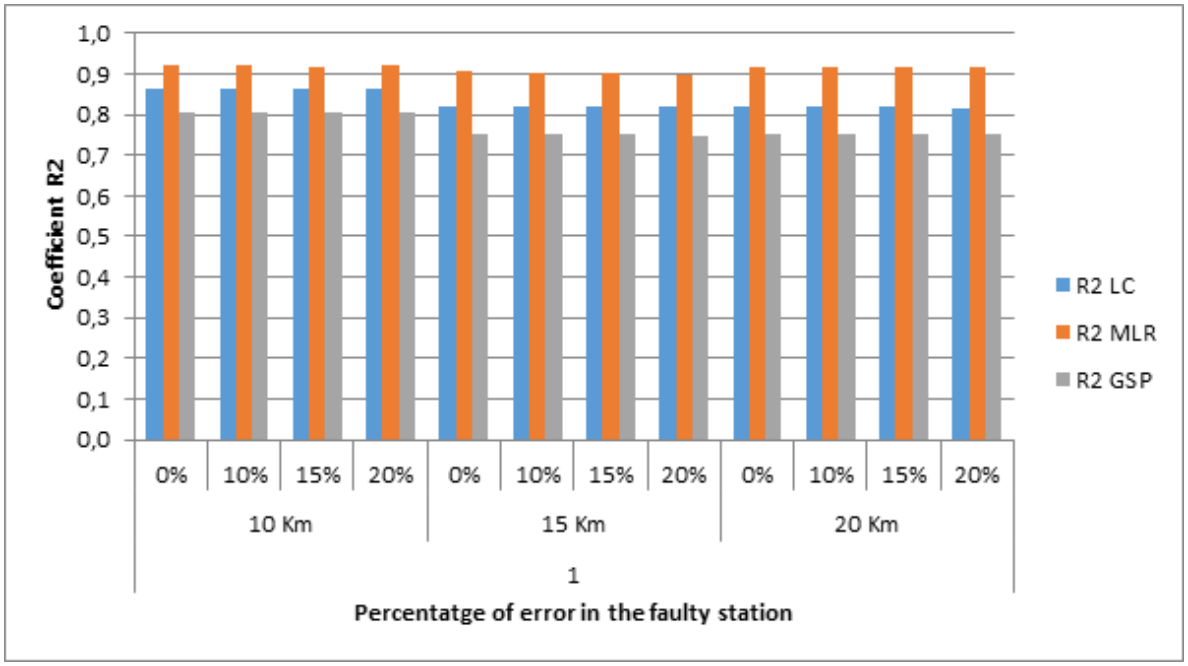


Figure 5.15: Comparison of R2 for threshold 10, 15, 20 Km with K = 1

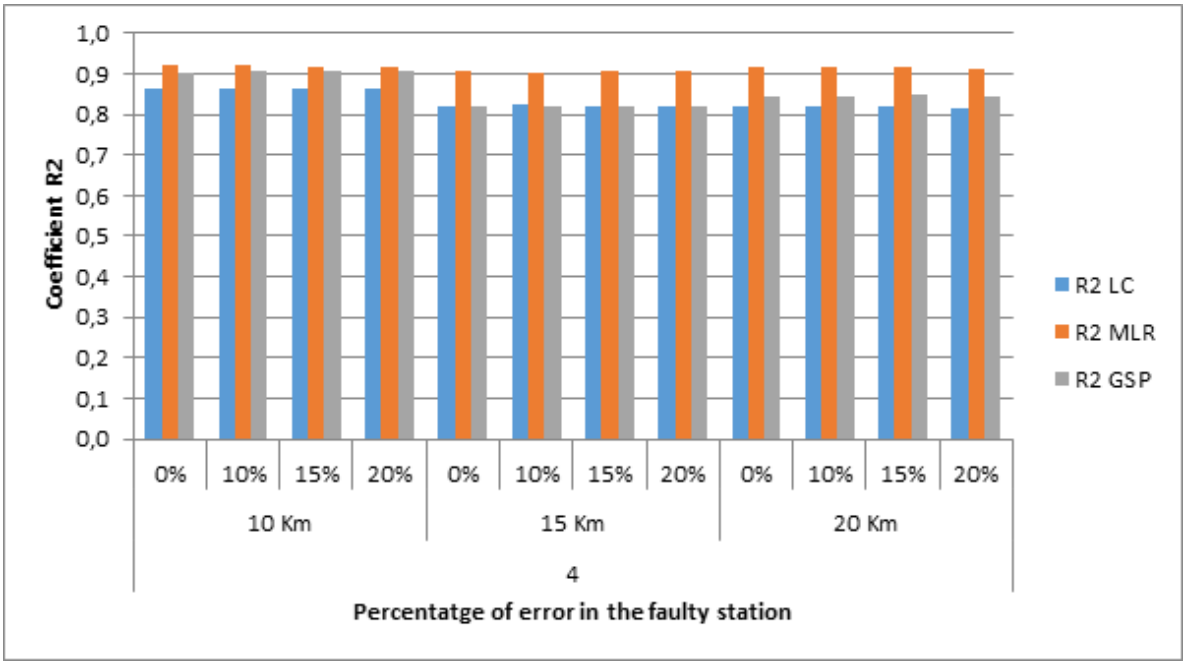


Figure 5.16: Comparison of R2 for threshold 10, 15, 20 Km with K = 4.

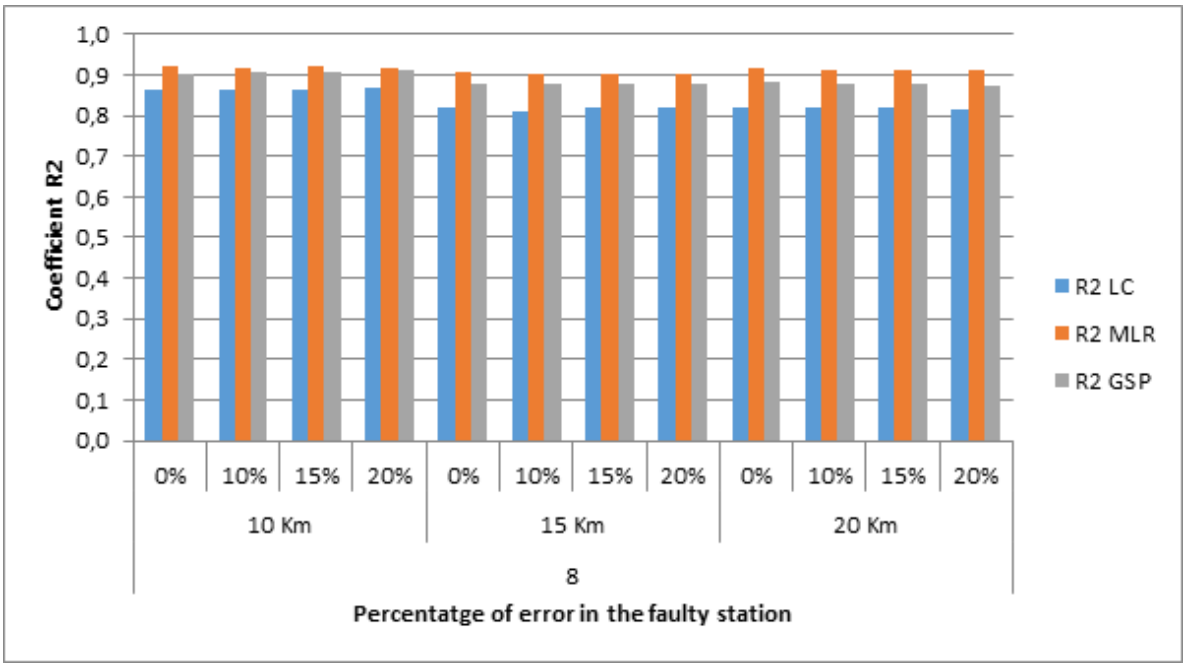


Figure 5.17: Comparison of R2 for threshold 10, 15, 20 Km with K = 8.

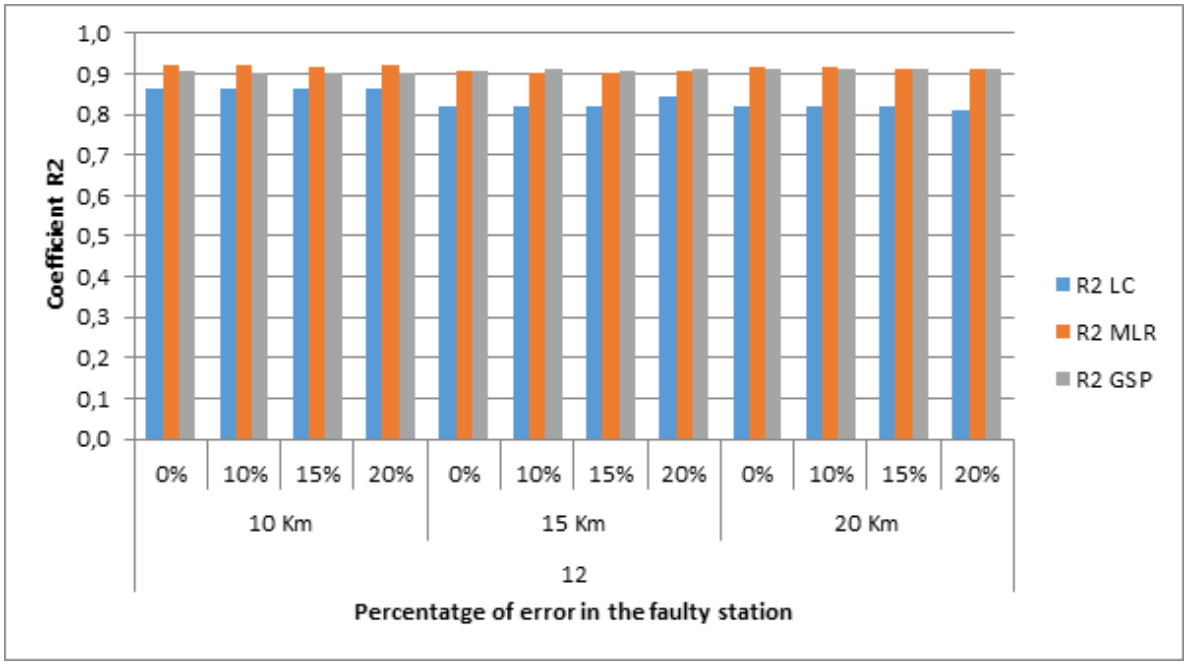


Figure 5.18: Comparison of R2 for threshold 10, 15, 20 Km with K = 8.

5.4 Pollutant PM_{10}

As with previous pollutants, this section presents the selected stations, the adjacencies between nodes, and the performances of the evaluation methods.

Figure 5.20 show the connections between nodes when using different distances, for values of 10, 15 and 20 Km.

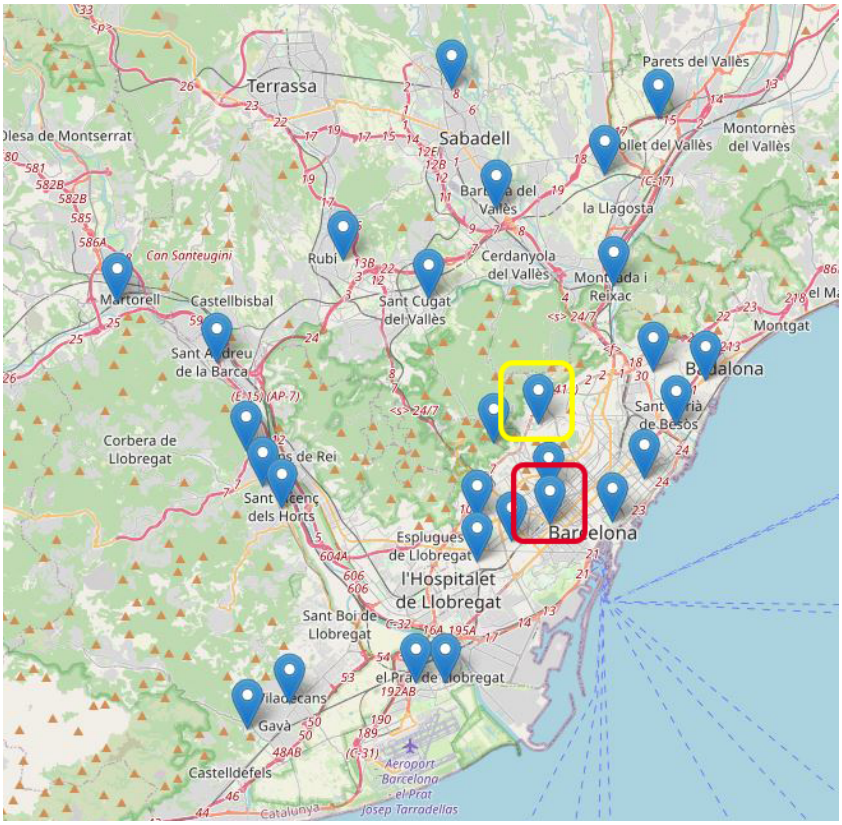
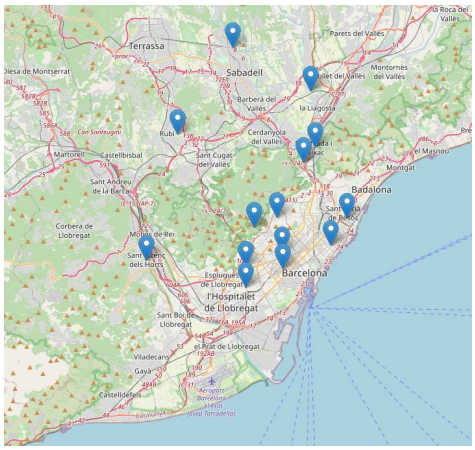


Figure 5.19: In red, the station to reconstruct the signal: Barcelona (Eixample). In yellow, the faulty station: Barcelona (Parc Vall Hebron).

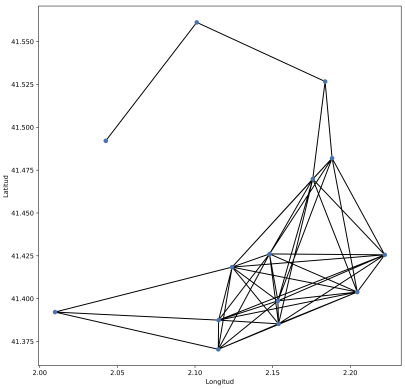
5.4.1 RMSE

In figure 5.21, *a priori*, we see the same behavior as the previous pollutants. Apparently, PM_{10} it is also a low-pass signal, where all three methods haver pretty much the same accuracy.

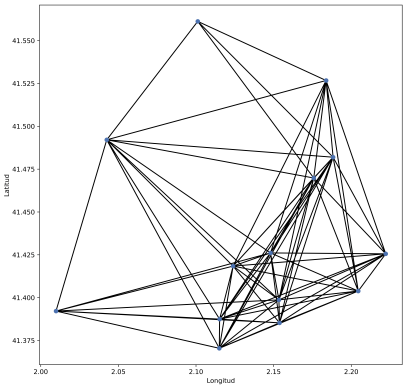
Figure 5.22 contains odd results. As the value of K and the threshold increases, the performance gets worse.



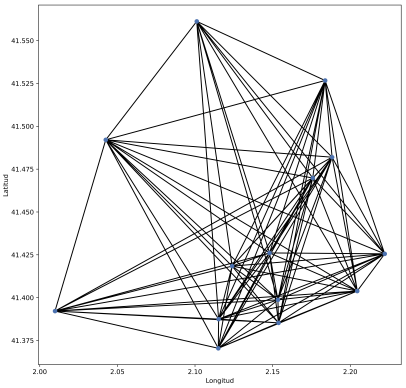
(a) PM_{10} sensor nodes in Barcelona.



(b) Max distance = 10 Km.



(c) Max distance = 15 Km.



(d) Max Distance = 20 Km.

Figure 5.20: Different cases of maximum distance between stations that measure PM_{10}.

When we think about PM_{10}, we know that they are inhalable particles, with diameters that are generally 10 micrometers and smaller, and normally come from construction sites or unpaved roads. Those particles are more focused around areas where someone is building something.

This makes us think that PM_{10} may not be a low-pass signal. In figure 5.23,it is possible to confirm that as the results got worse, as GSP has a terrible score on non low-pass signals, and hence, the performance drops.

This pollutant is not seasonal and it also depends on the industries and vehicles.

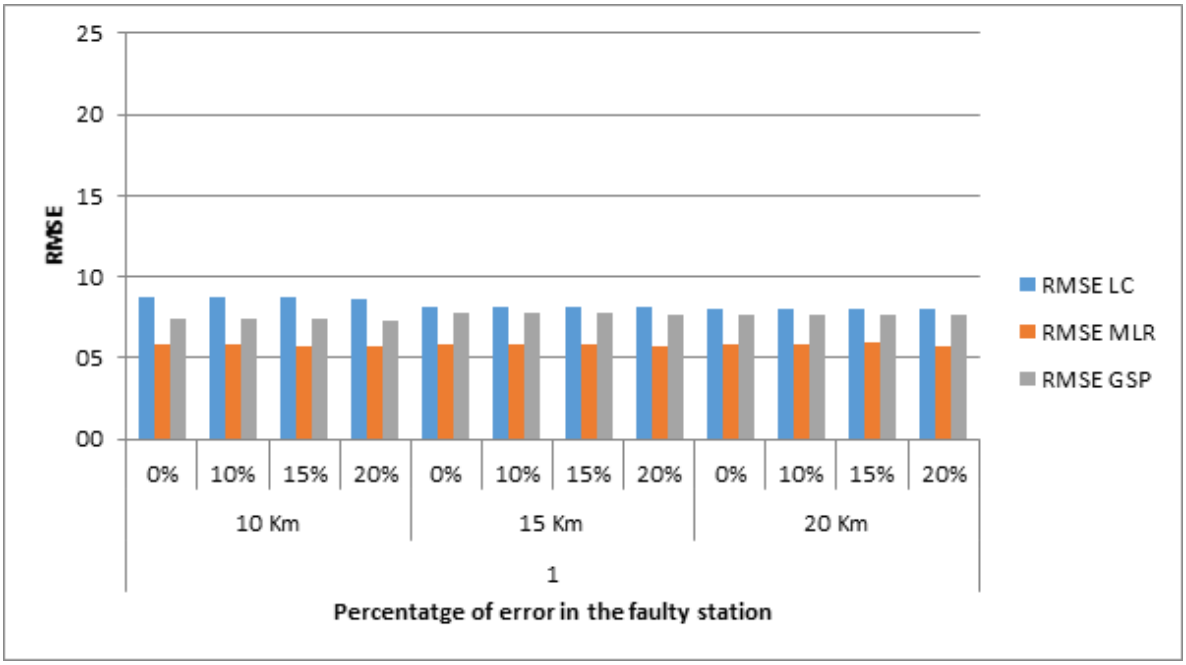


Figure 5.21: Comparison of RMSE for threshold 10, 15, 20 Km with K = 1.

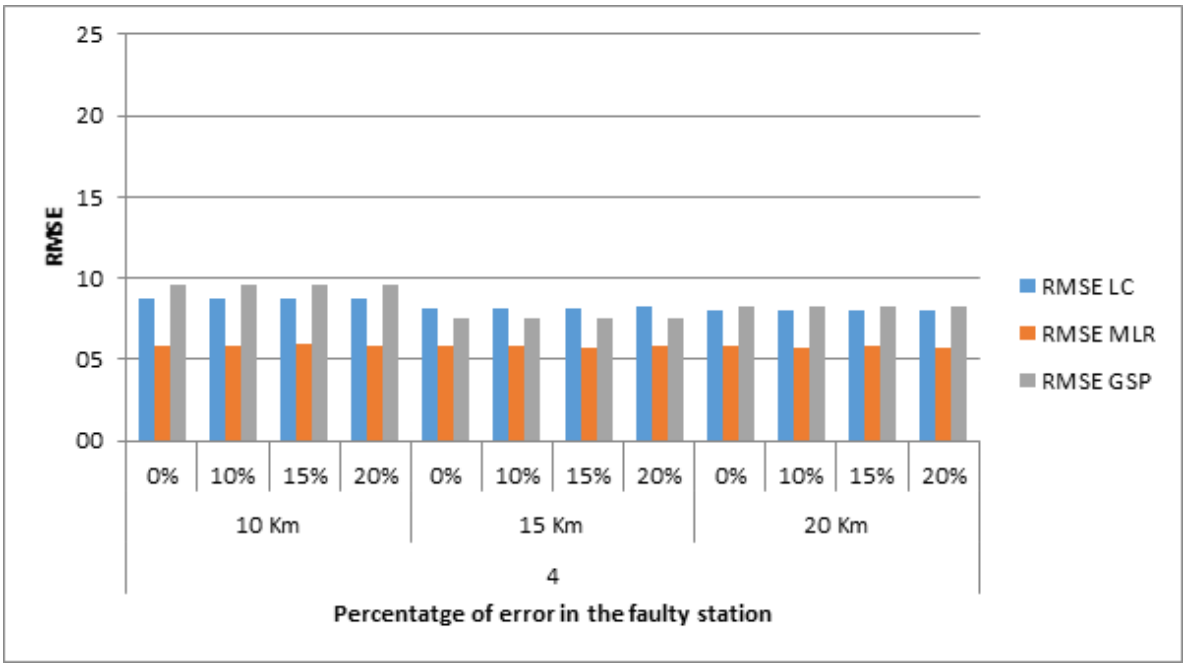


Figure 5.22: Comparison of RMSE for threshold 10, 15, 20 Km with K = 4.

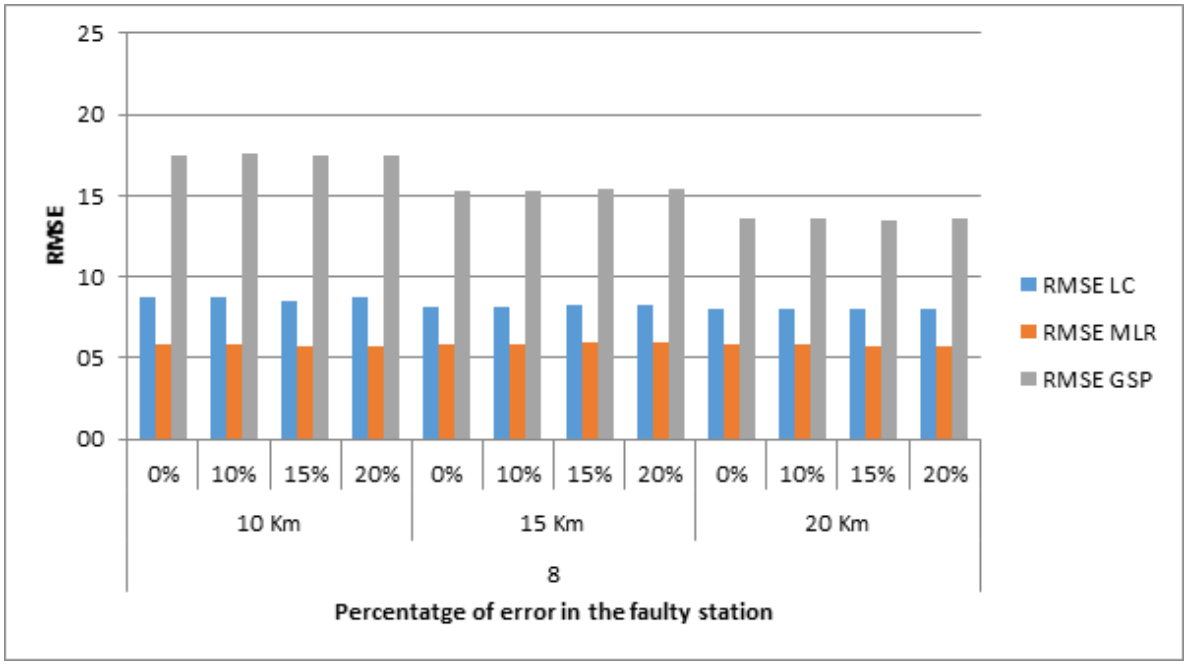


Figure 5.23: Comparison of RMSE for threshold 10, 15, 20 Km with K = 8.

5.4.2 R2

In figures 5.24, 5.25 and 5.26 it is shown the massive drop on GSP performance.

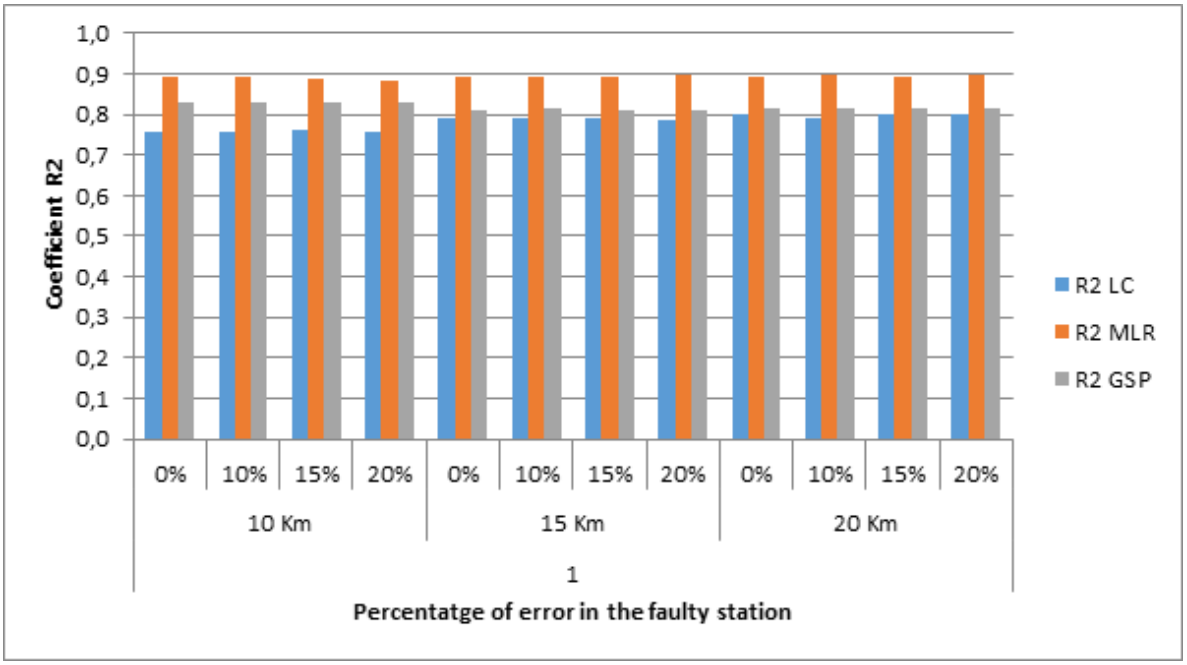


Figure 5.24: Comparison of R2 for threshold 10, 15, 20 Km with K = 1

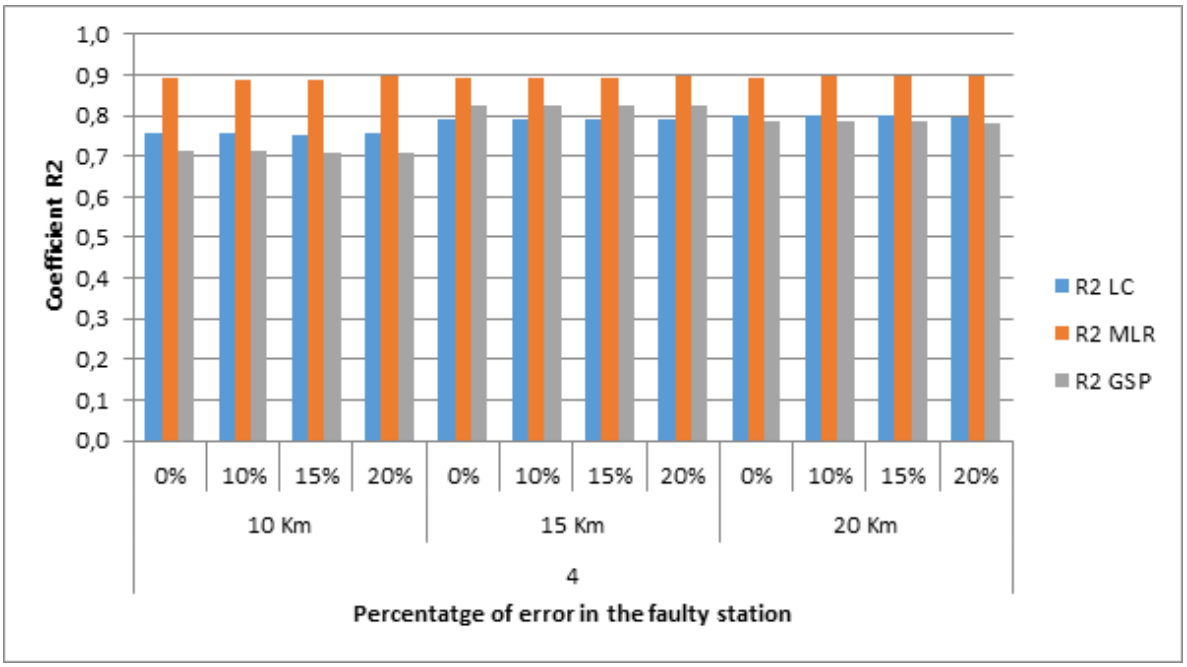


Figure 5.25: Comparison of R2 for threshold 10, 15, 20 Km with K = 4.

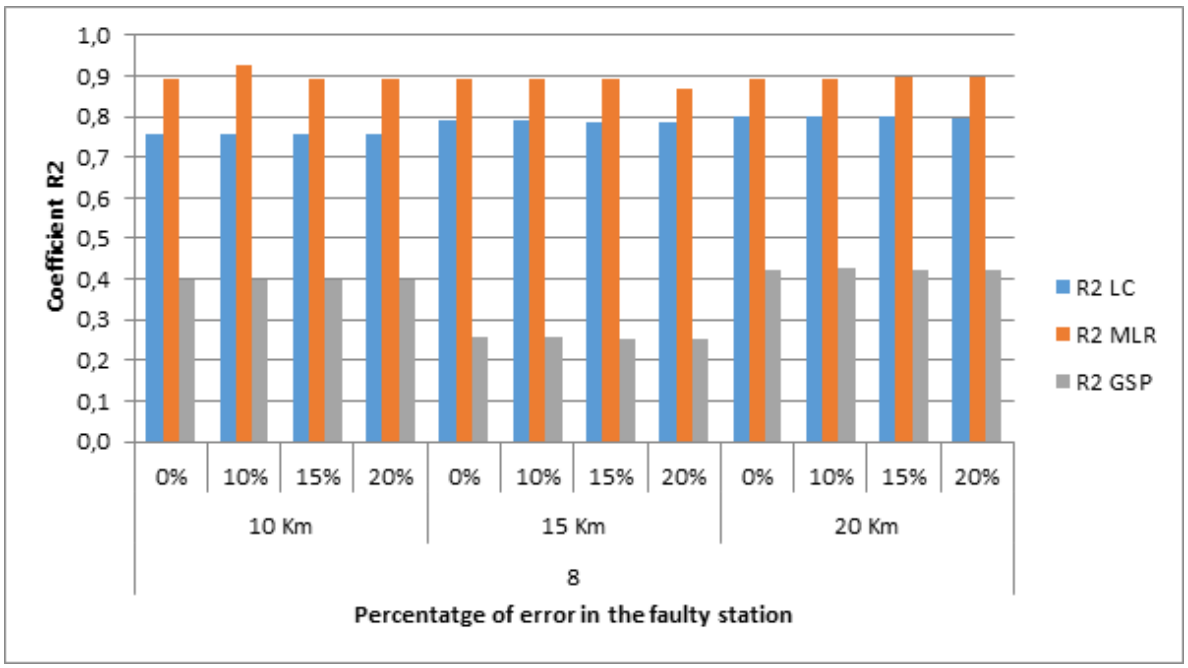


Figure 5.26: Comparison of R2 for threshold 10, 15, 20 Km with K = 8.

Chapter 6

Conclusion & Future Work

While traditional approaches for graph analysis consider only graph topology and spectral properties of the graph, when dealing with signals on graphs, it is important to consider both data and the topology. This unified approach defines and implements a better methods of analysis and reconstruction, as we have seen during this exploration.

In general, what we can conclude from the results, is that a coefficient of R2 below 0.6 points indicates that the models have a bad behavior (i.e. when using K = 1), and because the value of K impacts the number of nodes used in the reconstruction, we should aim for higher values.

Recall that $K \leq M < N$, where M is the minimum number of nodes to reconstruct the signal. That means that the indicator R2 gives us an idea when then number of nodes starts to be acceptable in the reconstruction method. For example, in the figure 5.8, where K = 8 and the maximum distance threshold is 20 Km, the coefficient R2 is between 0.7-0.8 and the RMSE from figure 5.5 is closer to the RMSE value from MLR.

It also reinforces the idea of looking for optimal Laplacian matrices that will build better graphs, since they will allow GSP to get a closer performance to MLR. It also shows that the GSP signal reconstruction method chooses the K nodes that have better smoothness (less signal variation).

The way the graph is constructed should be a method that takes into account smoothness, if GSP want to be successful. Otherwise you will have to use a large number of edges, i.e. a very large network, like the one we have with threshold = 20 Km, creating even more complex networks.

Regarding the cases where we have a station with errors, it seems that the network methods are robust in all threshold values, obtaining similar

coefficients for both RMSE and R2 coefficients through all cases, when a close station has faulty data.

As future work, this thesis can be extended and improved in several directions

- Experiment a bit more to check weather GSP is able to detect which station is the faulty one.
- Make a study on the months during the COVID-19 quarantine and compare them to previous year.
- Apply the same study on data from summer, as the values of O_3 depend on a reaction between sunlight and NO_x.

Bibliography

[1] World Health Organizacion. *WHO | World Health Organization.* URL: https://www.who.int/.

[2] United States Environmental Protection Agency. *Particular Matter (PM) Pollution.* 2020. URL: https://www.epa.gov/ground-level-ozone-pollution/ground-level-ozone-basics.

[3] United States Environmental Protection Agency. *Basic information about NO2.* 2020. URL: https://www.epa.gov/no2-pollution/basic-information-about-no2.

[4] United States Environmental Protection Agency. *Particular Matter (PM) Pollution.* 2020. URL: https://www.epa.gov/pm-pollution/particulate-matter-pm-basics.

[5] Temesegan Walelign Ayele and Rutvik J. Mehta. "Air pollution monitoring and prediction using IoT". In: *Second International Conference of Inventive Communication and Computational Technologies (ICICCT)* (2018), pp. 1713–1745.

[6] Dixian Zhy, Changjie Caie, Tianbao Yang, and Xun Zhou. "A Machine Learning Approach for Air Quality Prediction: Model Regularization and Optimization". In: (2018).

[7] Krzysztof Siwek and Stanislaw Osowski. "Data mining methods for prediction of air pollution". In: *International Journal of Applied Mathematics and Computer Science* 26 (2016), pp. 467–478.

[8] Ljubiša Stanković et al. "Graph Signal Processing - Part I: Graphs, Graph Spectra, and Spectral Clustering". In: (2019).

[9] Ljubiša Stanković et al. “Graph Signal Processing - Part II: Processing and Analyzing Signals on Graphs”. In: (2019).

[10] Ljubiša Stanković et al. “Graph Signal Processing - Part II: Processing and Analyzing Signals on Graphs”. In: (2020).

[11] A. V. Oppenheim and R. W. Shafer. *Digital Signal Processing.* Englewood Cliffs, New Jersey: Prentice-Hall, 1975.

[12] A. V. Oppenheim and R. W. Shafer. *Discrete-Time Signal Processing.* Englewood Cliffs, New Jersey: Prentice-Hall, 1989.

[13] P. Frossard , J. Kovačevič Ortega et al. “Graph Signal Processing: Overview, Challenges and Applications”. In: *Proceedings of the IEEE* 106, no. 5 (2018), pp. 808–828.

[14] G. B. Riberino and J. B. Lima. “Graph Signal Processing in a nutshell”. In: *Journal of Communication and Information systems* 33, no. 1 (2018).

[15] Michael Müller, Jonas Meyer, and Cristoph Hueglin. “Design of an ozone nitrogen dioxide sensor unit and its long-term operation within a sensor network in the city of Zurich”. In: *Atmospheric Measruement Techniques* 10 (2017), pp. 3783–3799.

[16] Y. Liu, K. Zhou and Y. Lei. “Using Bayesian inference framework towards identifying gas species and concentration from high temperature resistive sensor array data”. In: (2015).

[17] Balz Maag, Zimu Zhou, and Lothar Thiele. “A Survey on Sensor Calibration in Air Pollution Monitoring Deployments”. In: *IEEE Internet of Things Journal* 5 (2018), pp. 4857–4870.

[18] Pau Ferrer-Cid, José María Barceló-Ordinas, Joge García-Vidal, Anna Ripoll, and Mar Viana. “A comparative study of calibration methods for low-cost ozone sensors in IoT Platforms”. In: *IEEE Internet of Things* 6 (2019), pp. 9563–9571.

[19] Pau Ferrer-Cid, José María Barceló-Ordinas, Joge García-Vidal, Anna Ripoll, and Mar Viana. “Multisensor data fusion calibration in IoT air pollution platforms”. In: *IEEE Internet of Things* 7 (2020), pp. 3124–3132.

[20] Chun Feng Lin et al. "Evaluation and calibration of Aeroqual series 500 portable gas sensors for accurate measurement of ambient ozone and nitrogen dioxide". In: *Atmospheric Environment* 100 (2015), pp. 111–116.

[21] L. Spinelle, M. Gerboles, M. G. Villani, M. Aleixandre, and F. Bonavitacola. "Calibration of a cluster of low-cost sensors for the measurement of air pollution in ambient air". In: *IEEE SENSORS* (2014).

[22] Carl Malings et al. "Development of a general calibration model and long-term performance evaluation of low-cost sensors for air pollutant gas monitoring". In: *Atmospheric Measurement Techniques* 12 (2019), pp. 903–920.

[23] *CAPTOR project*. 2020. URL: https://www.captor-project.eu/en/.

[24] *Captor O3*. 2016. URL: https://www.youtube.com/watch?v=NsnFPStbcag (visited on 04/08/2020).

[25] *Ambient air ozone concentrations using metal-oxide low-cost sensors: Spain and Italy, summer 2017*. 2019. URL: https://zenodo.org/record/2564753#.XO5XP5wvDrE.

[26] *Ambient air ozone concentrations using metal-oxide low-cost sensors: Spain and Italy, summer 2018*. 2019. URL: https://zenodo.org/record/2564825#.XO5XYpwvDrE.

[27] *A comparative study of calibration methods for low-cost ozone sensors in IoT platforms*. 2019. URL: https://www.youtube.com/watch?v=NsnFPStbcag.

[28] *Statistical Analysis of Networks and Systems (SANS) Research Group*. 2020. URL: http://sans.ac.upc.edu/.

[29] Sami Kaivonen and Edith C.-H. Ngai. "Real-time air pollution monitoring with sensors on city bus". In: *Digital Communications and Networks* 6 (2020), pp. 23–30.

[30] *GreenIoT*. 2017. URL: http://user.it.uu.se/~eding810/GreenIoT/.

[31] Ed T. Bullmore and Olaf Sporns. "The economy of brain network organization". In: *Nature Reviews Neuroscience* 13 (2012), pp. 336–349.

[32] Olaf Sporns. "Networks of the Brain". In: 2010.

[33] R. Wagner, Hyeokho Choi, R. Baraniuk, and Véronique Delouille. "Distributed wavelet transform for irregular sensor network grids". In: *IEEE/SP 13th Workshop on Statistical Signal Processing, 2005* (2005), pp. 1196–1201.

[34] R. Wagner, R. Baraniuk, Sunwen Du, D. Johnson, and Arnaldo Cohen. "An architecture for distributed wavelet analysis and processing in sensor networks". In: *2006 5th International Conference on Information Processing in Sensor Networks* (2006), pp. 243–250.

[35] Godwin Shen and Antonio Ortega. "Joint Routing and 2D Transform Optimization for Irregular Sensor Network Grids Using Wavelet Lifting". In: *2008 International Conference on Information Processing in Sensor Networks (ipsn 2008)* (2008), pp. 183–194.

[36] Ireneusz Jabłoński. "Graph Signal Processing in Applications to Sensor Networks, Smart Grids, and Smart Cities". In: *IEEE Sensors Journal* 17 (2017), pp. 7659–7666.

[37] K. Benzi, V. Kalofolias, X. Bresson, and P. Vandergheynst. "Song recommendation with non-negative matrix factorization and graph total variation". In: *IEEE Int. Conf. Acoust. Speech Signal Process (ICA SSP)* (2016), pp. 2439–2443.

[38] M. Valko. "Spectral bandits for smooth graph functions". In: *Proc. 31st Int. Conf. Mach. Learn (ICML)* (2014), pp. 1205–12015.

[39] J. Masci, D. Boscaini, M. M. Bronstein, and P. Vandergheynst. "Geodesic convlutional neural networks on Rienmannian manifolds". In: *IEEE Int. Conf. Comput. Vis. Workshop (ICCVW)* (2015), pp. 832–840.

[40] J. Masci et al. "Learning class-specific descriptors for deformable shapes using localized spectral convolutional networks". In: *Proc. Eurpgraph. Symp. Geometry Process* (2015), pp. 1–11.

[41] Generalitat de Catalunya. *Dades Obertes de Catalunya*. URL: https://analisi.transparenciacatalunya.cat.

[42] EPFL LTS2. *PyGSP: Graph Signal Processing in Python*. URL: https://pygsp.readthedocs.io/en/stable/.